

Recursive Sampling for the Nyström Method

Cameron Musco
MIT
cnmusco@mit.edu

Christopher Musco
MIT
cpmusco@mit.edu

March 17, 2017

Abstract

We give the first algorithm for kernel Nyström approximation that runs in *linear time in the number of training points* and is provably accurate for all kernel matrices, without dependence on regularity or incoherence conditions. The algorithm projects the kernel onto a set of s landmark points sampled by their *ridge leverage scores*, requiring just $O(ns)$ kernel evaluations and $O(ns^2)$ additional runtime. While leverage score sampling has long been known to give strong theoretical guarantees for Nyström approximation, by employing a fast recursive sampling scheme, our algorithm is the first to make the approach scalable. Empirically we show that it finds more accurate, lower rank kernel approximations in less time than popular techniques such as uniformly sampled Nyström approximation and the random Fourier features method.

1 Introduction

The kernel method is a powerful tool that allows standard linear learning and prediction algorithms (SVMs, linear regression, etc.) to be applied to nonlinear problems. The key idea is to map data to a higher dimensional *kernel feature space*, such that linear relationships in this space correspond to nonlinear relationships in the original data.

Typically this mapping is implicit. A *kernel function* is used to compute inner products in the high-dimensional kernel space, without ever actually mapping original data points to the space. Given n data points $\mathbf{x}_1, \dots, \mathbf{x}_n$, the $n \times n$ kernel matrix \mathbf{K} is formed where $\mathbf{K}_{i,j}$ contains the high-dimensional inner product between \mathbf{x}_i and \mathbf{x}_j , as computed by the kernel function. All computations required by a linear learning method are performed using the inner product information in \mathbf{K} .

Unfortunately, the transition from linear to nonlinear comes at a high cost. Just generating the entries of \mathbf{K} requires $\Theta(n^2)$ time, which is prohibitive for large datasets.

1.1 Kernel approximation

A large body of work seeks to accelerate kernel methods by finding a compressed, often low-rank, approximation $\tilde{\mathbf{K}}$ to the true kernel matrix \mathbf{K} . Techniques include random sampling and embedding [AMS01, BBV06, ANW14], *random Fourier feature* methods for shift invariant kernels [RR07, RR09, LSS13], and incomplete Cholesky factorization [FS02, BJ02].

One of the most well studied techniques is the *Nystrom method*, which constructs $\tilde{\mathbf{K}}$ by projecting \mathbf{K} onto a subset of “landmark” data points [WS01]. Once s data points are selected, $\tilde{\mathbf{K}}$ (in factored form) takes just $O(ns)$ kernel evaluations and $O(s^3)$ additional time to compute, requires $O(ns)$ space to store, and can be manipulated quickly in downstream applications. For example, inverting $\tilde{\mathbf{K}}$ for kernel regression takes $O(ns^2)$ time.

The Nystrom method performs well in practice [YLM⁺12, GM13, TRVR16], is widely implemented [HFH⁺09, PVG⁺11, IBM14], and is used in a number of applications under different names such as “landmark isomap” [DST03] and “landmark MDS” [Pla05]. In the classic variant, landmark points are selected uniformly at random. However, significant research seeks to improve performance via data-dependent sampling approaches that select landmarks which more closely approximate the full kernel matrix than uniformly sampled landmarks [SS00, DM05, ZTK08, BW09, KMT12, WZ13, GM13, LJS16].

Theoretical work has converged on *leverage score* based approaches, as they give the strongest provable guarantees for both kernel approximation [DMM08, GM13] and statistical performance in downstream applications [AM15, RCR15, Wan16]. Leverage scores capture how important an individual data point is in composing the span of the kernel matrix.

Unfortunately, these scores are prohibitively expensive to compute, and while a number of approximation schemes exist [DMIMW12, GM13, AM15, CLV16], all require at least $\Omega(n^2)$ time or only run quickly under strong regularity conditions on \mathbf{K} (e.g. good conditioning or data “incoherence”). Hence, leverage score based approaches for kernel approximation remain largely in the domain of theory, with limited practical impact [KMT12, LBKL15, YPW15].

1.2 Our contributions

In this work, we close the gap between strong approximation guarantees and computational efficiency for kernel approximation. We present a new Nystrom algorithm based on *recursive leverage*

score sampling which achieves the “best of both worlds”: it obtains kernel approximations matching the high accuracy of other leverage score methods while only requiring $O(ns)$ kernel evaluations and $O(ns^2)$ computation time for s landmark points.

Theoretically, this runtime is surprising. In the typical case when $s \ll n$, the algorithm evaluates just a small subset of \mathbf{K} , ignoring most of the kernel space inner products. Yet its performance guarantees hold for general kernels, requiring *no assumptions on coherence or regularity*.

Empirically, the runtime’s linear dependence on n means that our method is the first leverage score algorithm that can compete with the most commonly implemented techniques, including the classic uniform sampling Nyström method and random Fourier features sampling [RR07]. Since our algorithm obtains higher quality samples, we show experimentally that it outperforms these methods on benchmark datasets – it can obtain as accurate a kernel approximation in significantly less time. As a bonus, our approximations have lower rank, so they can be stored in less space and processed more quickly in downstream learning tasks.

1.3 Paper outline

Our recursive sampling algorithm is built on top of a Nyström scheme of Alaoui and Mahoney that samples landmark points based on their *ridge leverage scores* [AM15]. After reviewing preliminaries in Section 2, in Section 3 we analyze this scheme, which we refer to as *RLS-Nyström*. To simplify prior work, which studies the statistical performance of RLS-Nyström for specific kernel learning tasks [AM15, RCR15, Wan16], we prove a strong, application independent approximation guarantee: for any λ , if $\tilde{\mathbf{K}}$ is constructed with $s = \Theta(d_{\text{eff}}^\lambda \log d_{\text{eff}}^\lambda)$ samples¹, where $d_{\text{eff}}^\lambda = \text{tr}(\mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1})$ is the so-called “ λ -effective dimensionality” of \mathbf{K} , then with high probability:

$$\|\mathbf{K} - \tilde{\mathbf{K}}\|_2 \leq \lambda.$$

In Appendix C, we show that this guarantee implies prior results on the statistical performance of RLS-Nyström for kernel ridge regression and canonical correlation analysis. We also use it to prove new results on the performance of RLS-Nyström for kernel rank- k PCA and k -means clustering – in both cases just $O(k \log k)$ samples are required to obtain a solution with good accuracy.

After affirming the favorable theoretical properties of RLS-Nyström, in Section 4 we show that its runtime can be significantly improved using a recursive sampling approach. Intuitively our algorithm is simple. We show how to approximate the kernel ridge leverage scores using a *uniform* sample of $\frac{1}{2}$ of our input points. While the subsampled kernel matrix still has a prohibitive $n^2/4$ entries, we can *recursively approximate* it, using our same sampling algorithm. If our final Nyström approximation will use s landmarks, the recursive approximation only needs rank $O(s)$, which lets us estimate the ridge leverage scores of the original kernel matrix in just $O(ns^2)$ time. Since n is cut in half at each level of recursion, our total runtime is $O\left(ns^2 + \frac{ns^2}{2} + \frac{ns^2}{4} + \dots\right) = O(ns^2)$, significantly improving upon the method of [AM15], which takes $\Theta(n^3)$ time in the worst case.

Our approach builds on recent work on iterative sampling methods for approximate linear algebra [CLM⁺15, CMM17]. While the analysis in the kernel setting is technical, our final algorithm is simple and easy to implement. We present and test a parameter-free variation of Recursive RLS-Nyström in Section 5. We confirm that it scales to very large datasets and demonstrate superior performance in kernel approximation.

¹ $O(d_{\text{eff}}^\lambda \log d_{\text{eff}}^\lambda)$ samples is within a log factor of the best possible for any low-rank approximation with error λ .

2 Preliminaries

Consider an input space \mathcal{X} and a positive semidefinite kernel $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$. Let \mathcal{F} be an associated reproducing kernel Hilbert space and $\phi : \mathcal{X} \rightarrow \mathcal{F}$ be a (typically nonlinear) feature map such that for any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, $K(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle_{\mathcal{F}}$. Given a set of n input points $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$, define the kernel matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ by $\mathbf{K}_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$.

It will often be natural to consider the kernelized data matrix that generates \mathbf{K} . Informally, let $\Phi \in \mathbb{R}^{n \times d'}$ be the matrix containing $\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_n)$ as its rows (note that d' may be infinite). $\mathbf{K} = \Phi \Phi^T$. While we use Φ for intuition, in our formal proofs we replace it with any matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$ satisfying $\mathbf{B} \mathbf{B}^T = \mathbf{K}$ (e.g. a Cholesky factor).

We repeatedly use the singular value decomposition, which allows us to write any rank r matrix $\mathbf{M} \in \mathbb{R}^{n \times d}$ as $\mathbf{M} = \mathbf{U} \Sigma \mathbf{V}^T$, where $\mathbf{U} \in \mathbb{R}^{n \times r}$ and $\mathbf{V} \in \mathbb{R}^{d \times r}$ have orthogonal columns (the left and right singular vectors of \mathbf{M}), and $\Sigma \in \mathbb{R}^{r \times r}$ is a positive diagonal matrix containing the singular values: $\sigma_1(\mathbf{M}) \geq \sigma_2(\mathbf{M}) \geq \dots \geq \sigma_r(\mathbf{M})$. \mathbf{M} 's pseudoinverse is given by $\mathbf{M}^+ = \mathbf{V} \Sigma^{-1} \mathbf{U}^T$.

2.1 Nyström approximation

The Nyström method selects a subset of “landmark” points and uses them to construct a low-rank approximation to \mathbf{K} . Given a matrix $\mathbf{S} \in \mathbb{R}^{n \times s}$ that has a single entry in each column equal to 1 so that $\mathbf{K} \mathbf{S}$ is a subset of s columns from \mathbf{K} , the associated Nyström approximation is:

$$\tilde{\mathbf{K}} = \mathbf{K} \mathbf{S} (\mathbf{S}^T \mathbf{K} \mathbf{S})^+ \mathbf{S}^T \mathbf{K}. \quad (1)$$

$\tilde{\mathbf{K}}$ can be stored in $O(ns)$ space by separately storing $\mathbf{K} \mathbf{S} \in \mathbb{R}^{n \times s}$ and $(\mathbf{S}^T \mathbf{K} \mathbf{S})^+ \in \mathbb{R}^{s \times s}$. Furthermore, the factors can be computed using just $O(ns)$ evaluations of the kernel inner product to form $\mathbf{K} \mathbf{S}$ and $O(s^3)$ time to compute $(\mathbf{S}^T \mathbf{K} \mathbf{S})^+$. Typically $s \ll n$ so these costs are significantly lower than the cost to form and store the full kernel matrix \mathbf{K} .

We view Nyström approximation as a low-rank approximation to the dataset in feature space. Recalling that $\mathbf{K} = \Phi \Phi^T$, \mathbf{S} selects s kernelized data points $\mathbf{S}^T \Phi$ and we approximate Φ using its projection onto these points. Informally, let $\mathbf{P}_{\mathbf{S}} \in \mathbb{R}^{d' \times d'}$ be the orthogonal projection onto the row span of $\mathbf{S}^T \Phi$. We approximate Φ by $\tilde{\Phi} \stackrel{\text{def}}{=} \Phi \mathbf{P}_{\mathbf{S}}$. We can write $\mathbf{P}_{\mathbf{S}} = \Phi^T \mathbf{S} (\mathbf{S}^T \Phi \Phi^T \mathbf{S})^+ \mathbf{S}^T \Phi$. Since it is an orthogonal projection, $\mathbf{P}_{\mathbf{S}} \mathbf{P}_{\mathbf{S}}^T = \mathbf{P}_{\mathbf{S}}^2 = \mathbf{P}_{\mathbf{S}}$, and so we can write:

$$\tilde{\mathbf{K}} = \tilde{\Phi} \tilde{\Phi}^T = \Phi \mathbf{P}_{\mathbf{S}}^2 \Phi^T = \Phi (\Phi^T \mathbf{S} (\mathbf{S}^T \Phi \Phi^T \mathbf{S})^+ \mathbf{S}^T \Phi) \Phi^T = \mathbf{K} \mathbf{S} (\mathbf{S}^T \mathbf{K} \mathbf{S})^+ \mathbf{S}^T \mathbf{K}.$$

This recovers the standard Nyström approximation (1). Note that the above view is presented for intuition – we do not rigorously handle possibly infinite dimensional feature spaces. To make the argument formal, replace Φ with any $\mathbf{B} \in \mathbb{R}^{n \times n}$ satisfying $\mathbf{B} \mathbf{B}^T = \mathbf{K}$. Such a \mathbf{B} is guaranteed to exist since \mathbf{K} is positive semidefinite.

3 The RLS-Nyström method

The RLS-Nyström method of [AM15] uses ridge leverage score sampling to select landmark data points. Our main algorithmic contribution is showing how to perform this sampling in just $O(ns^2)$ time for s samples. However, before any runtime considerations, we first introduce the method and show that it satisfies strong approximation guarantees for any kernel matrix \mathbf{K} .

3.1 Ridge leverage scores

In classical Nyström approximation (1), \mathbf{S} is formed by sampling data points uniformly at random. Uniform sampling can work in practice, but it only gives theoretical guarantees under strong regularity or incoherence assumptions on \mathbf{K} [Git11]. It will fail for many natural kernel matrices where the relative “importance” of points is not uniform across the dataset

For example, imagine a dataset where points fall into several clusters, but one of the clusters is much larger than the rest. Uniform sampling will tend to oversample landmarks from the large cluster while undersampling or possibly missing smaller but still important clusters. Approximation of \mathbf{K} and learning performance (e.g. classification accuracy) will decline as a result.

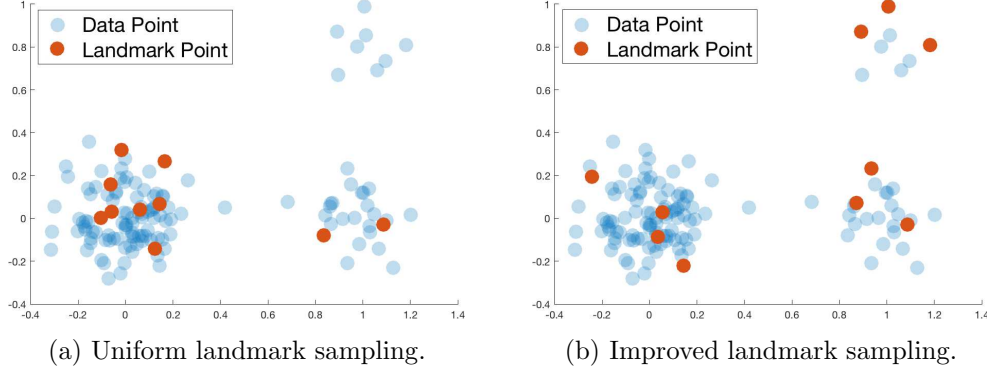


Figure 1: Uniform sampling for Nyström approximation can oversample from denser parts of the dataset. A better Nyström scheme will select points that more equally cover the relevant data.

To combat this issue, alternative methods compute a measure of point importance that is used to select landmarks. For example, one heuristic applies k -means clustering to the input and takes the cluster centers as landmarks [ZTK08]. A large body of theoretical work measures importance using variations on the *statistical leverage scores*. One natural variation is the *ridge leverage score*:

Definition 1 (Ridge leverage scores [AM15]). *For any $\lambda > 0$, the λ -ridge leverage score of data point \mathbf{x}_i with respect to the kernel matrix \mathbf{K} is defined as*

$$l_i^\lambda(\mathbf{K}) \stackrel{\text{def}}{=} (\mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1})_{i,i}, \quad (2)$$

For any $\mathbf{B} \in \mathbb{R}^{n \times n}$ satisfying $\mathbf{B}\mathbf{B}^T = \mathbf{K}$, we can also write

$$l_i^\lambda(\mathbf{K}) = \mathbf{b}_i^T (\mathbf{B}^T \mathbf{B} + \lambda\mathbf{I})^{-1} \mathbf{b}_i, \quad (3)$$

where $\mathbf{b}_i^T \in \mathbb{R}^{1 \times n}$ is the i^{th} row of \mathbf{B} .

Above \mathbf{I} refers to the $n \times n$ identity matrix. For conciseness we write $l_i^\lambda(\mathbf{K})$ as l_i^λ and include the argument only when referring to the ridge leverage scores of a kernel matrix other than \mathbf{K} . To check that (2) and (3) are equivalent note that $\mathbf{b}_i^T (\mathbf{B}^T \mathbf{B} + \lambda\mathbf{I})^{-1} \mathbf{b}_i = (\mathbf{B}(\mathbf{B}^T \mathbf{B} + \lambda\mathbf{I})^{-1} \mathbf{B}^T)_{i,i}$. Using the SVD to write $\mathbf{B} = \mathbf{U}\Sigma\mathbf{V}^T$ and accordingly $\mathbf{K} = \mathbf{U}\Sigma^2\mathbf{U}^T$ confirms that $\mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1} = \mathbf{B}(\mathbf{B}^T \mathbf{B} + \lambda\mathbf{I})^{-1} \mathbf{B}^T = \mathbf{U}\Sigma^2 (\Sigma^2 + \lambda\mathbf{I})^{-1} \mathbf{U}^T$.

It’s not hard to check (see [CLM⁺15]) that the ridge scores can be defined alternatively as:

$$l_i^\lambda = \min_{\mathbf{y} \in \mathbb{R}^n} \frac{1}{\lambda} \|\mathbf{b}_i^T - \mathbf{y}^T \mathbf{B}\|_2^2 + \|\mathbf{y}\|_2^2. \quad (4)$$

This formulation provides better insight into the meaning of these scores. Since $\mathbf{B}\mathbf{B}^T = \mathbf{K}$, any kernel learning algorithm effectively performs linear learning with \mathbf{B} ’s rows as data points. So the ridge scores should reflect the relative importance or uniqueness of these rows. From (4) it’s clear that $l_i^\lambda \leq 1$ since we can set \mathbf{y} to the i^{th} standard basis vector. A row \mathbf{b}_i^T will have ridge score $\ll 1$ (i.e. is less important) when it’s possible to find a more “spread out” \mathbf{y} that uses other rows in \mathbf{B} to approximately reconstruct \mathbf{b}_i^T – in other words when the row is less unique.

3.2 Sum of ridge leverage scores

As is standard in leverage score methods, we don’t directly select landmarks to be the points with the highest scores. Instead, we sample each point with probability proportional to l_i^λ . I.e. if a point has the highest possible ridge leverage score of 1, we will select it with probability 1 to be a landmark. If a point has leverage score 1/100, we select it with probability 1/100.²

Accordingly, the number of landmarks selected, which controls $\tilde{\mathbf{K}}$ ’s rank, is a random variable with expectation equal to the *sum of the λ -ridge leverage scores*. To ensure compact kernel approximations, we want this sum to be small. Immediately from Definition 1, we have:

Fact 2 (Ridge leverage scores sum to the effective dimension).

$$\sum_{i=1}^n l_i^\lambda(\mathbf{K}) = \text{tr}(\mathbf{K}(\mathbf{K} + \lambda \mathbf{I})^{-1}). \quad (5)$$

$\text{tr}(\mathbf{K}(\mathbf{K} + \lambda \mathbf{I})^{-1})$ is a natural quantity, referred to as the “effective dimension” or “degrees of freedom” for a ridge regression problem on \mathbf{K} with regularization λ [HTF02]. We use the notation:

$$d_{\text{eff}}^\lambda \stackrel{\text{def}}{=} \text{tr}(\mathbf{K}(\mathbf{K} + \lambda \mathbf{I})^{-1}). \quad (6)$$

d_{eff}^λ increases monotonically as λ decreases. For any fixed λ it is essentially the smallest possible rank achievable for $\tilde{\mathbf{K}}$ satisfying the approximation guarantee given by RLS-Nyström: $\|\mathbf{K} - \tilde{\mathbf{K}}\|_2 < \lambda$.

3.3 The basic sampling algorithm

We can now introduce the RLS-Nyström method of Alaoui and Mahoney as Algorithm 1. Our pseudocode allows sampling each point by *any probability greater than l_i^λ* . This is useful later when we compute ridge leverage scores approximately. Naturally, oversampling landmarks can only improve $\tilde{\mathbf{K}}$ ’s accuracy. It could cause us to take more samples, but we will always ensure that the sum of our approximate ridge leverage scores is not much higher than that of the exact scores.

²To ensure concentration in our sampling algorithm, we will actually take points with probability ql_i^λ where q is a small oversampling parameter.

Algorithm 1 RLS-NYSTRÖM SAMPLING

input: $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$, kernel matrix \mathbf{K} , ridge parameter $\lambda > 0$, failure probability $\delta \in (0, 1/8)$

output: kernel approximation $\tilde{\mathbf{K}}$

- 1: Compute an over-approximation, $\tilde{l}_i^\lambda > l_i^\lambda$ for the λ -ridge leverage score of each $\mathbf{x}_1, \dots, \mathbf{x}_n$
 - 2: Set $p_i := \min \left\{ 1, \tilde{l}_i^\lambda \cdot 16 \log(\sum \tilde{l}_i^\lambda / \delta) \right\}$.
 - 3: Construct $\mathbf{S} \in \mathbb{R}^{n \times s}$ by sampling $\mathbf{x}_1, \dots, \mathbf{x}_n$ each independently with probability p_i . In other words, for each i add a column to \mathbf{S} with a 1 in position i with probability p_i .
 - 4: Form the Nyström approximation $\tilde{\mathbf{K}} := \mathbf{K}\mathbf{S}(\mathbf{S}^T\mathbf{K}\mathbf{S})^+\mathbf{S}^T\mathbf{K}$.
-

Note that an implementation of RLS-Nyström Sampling would not form $\tilde{\mathbf{K}}$ explicitly in Step 4, as this would take space and time quadratic in n . It would simply return $\mathbf{K}\mathbf{S} \in \mathbb{R}^{n \times s}$ along with $(\mathbf{S}^T\mathbf{K}\mathbf{S})^+ \in \mathbb{R}^{s \times s}$. Any kernel learning method can then access $\tilde{\mathbf{K}}$ implicitly. For example, the kernel method can be implemented as a linear method run on the $n \times s$ matrix $\mathbf{K}\mathbf{S}(\mathbf{S}^T\mathbf{K}\mathbf{S})^{+/2}$ whose rows serves as a compression of the data points in kernel space

3.4 Accuracy bounds

Like other leverage scores methods, RLS-Nyström sampling is appealing because it provably approximates any kernel matrix. In particular, we show that the algorithm produces a $\tilde{\mathbf{K}}$ which spectrally approximates \mathbf{K} up to a small additive error. This is the strongest type of approximation offered by any known Nyström method [GM13] and, importantly, it guarantees that $\tilde{\mathbf{K}}$ will provide provable accuracy when used in place of \mathbf{K} in many downstream machine learning applications.

Theorem 3 (Spectral error kernel approximation). *For any $\lambda > 0$ and $\delta \in (0, 1/8)$, RLS-Nyström (Algorithm 1) returns an $\mathbf{S} \in \mathbb{R}^{n \times s}$ such that with probability $1 - \delta$, $s \leq 2 \sum_i p_i$ and $\tilde{\mathbf{K}}$ satisfies:*

$$\tilde{\mathbf{K}} \preceq \mathbf{K} \preceq \tilde{\mathbf{K}} + \lambda \mathbf{I}. \quad (7)$$

When ridge scores are computed exactly, $\sum_i p_i = O\left(d_{\text{eff}}^\lambda \log(d_{\text{eff}}^\lambda / \delta)\right)$.

\preceq denotes the standard Loewner matrix ordering on positive semi-definite matrices³. Note that (7) immediately implies the well studied (see e.g [GM13]) spectral norm guarantee, $\|\mathbf{K} - \tilde{\mathbf{K}}\|_2 \leq \lambda$.

Intuitively, Theorem 3 guarantees that the $\tilde{\mathbf{K}}$ produced by RLS-Nyström well approximates the top of \mathbf{K} 's spectrum (i.e. any eigenvalues $> \lambda$) while allowing it to lose information about smaller eigenvalues, which are less important for many learning tasks.

Proof. It is clear from the view of Nyström approximation as a low-rank projection of the kernelized data (see Section 2.1) that $\tilde{\mathbf{K}} \preceq \mathbf{K}$. Formally, for any $\mathbf{B} \in \mathbb{R}^{n \times n}$ with $\mathbf{B}\mathbf{B}^T = \mathbf{K}$:

$$\tilde{\mathbf{K}} = \mathbf{K}\mathbf{S}(\mathbf{S}^T\mathbf{K}\mathbf{S})^+\mathbf{S}^T\mathbf{K} = \mathbf{B}\mathbf{P}_\mathbf{S}\mathbf{B}^T,$$

where $\mathbf{P}_\mathbf{S} = \mathbf{B}^T\mathbf{S}(\mathbf{S}^T\mathbf{B}\mathbf{B}^T\mathbf{S})^+\mathbf{S}^T\mathbf{B}$ is the orthogonal projection onto the row span of $\mathbf{S}^T\mathbf{B}$. Since $\mathbf{P}_\mathbf{S}$ is a projection $\|\mathbf{P}_\mathbf{S}\|_2 \leq 1$. So, for any $\mathbf{x} \in \mathbb{R}^n$:

$$\mathbf{x}^T\tilde{\mathbf{K}}\mathbf{x} = \mathbf{x}^T\mathbf{B}\mathbf{P}_\mathbf{S}\mathbf{B}\mathbf{x} = \|\mathbf{P}_\mathbf{S}\mathbf{B}\mathbf{x}\|_2^2 \leq \|\mathbf{B}\mathbf{x}\|_2^2 = \mathbf{x}^T\mathbf{K}\mathbf{x},$$

³ $\mathbf{M} \preceq \mathbf{N}$ means that $\mathbf{N} - \mathbf{M}$ is positive semidefinite.

which is equivalent to $\tilde{\mathbf{K}} \preceq \mathbf{K}$. It remains to show that $\mathbf{K} \preceq \tilde{\mathbf{K}} + \lambda \mathbf{I}$.

In Lemma 11, Appendix A, we apply a matrix Bernstein bound [Tro15] to prove that, when \mathbf{S} 's columns are reweighted by the inverse of their sampling probabilities, with probability $1 - \delta/2$:

$$\frac{1}{2} (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I}) \preceq \mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda \mathbf{I} \preceq \frac{3}{2} (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I}).$$

It is not hard to show (Corollary 13, Appendix A) that even if \mathbf{S} is unweighted, as in Algorithm 1, this bound implies the existence of some finite scaling factor $C > 0$ such that:

$$\mathbf{B}^T \mathbf{B} \preceq C \cdot \mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda \mathbf{I}. \quad (8)$$

Let $\bar{\mathbf{P}}_{\mathbf{S}} = \mathbf{I} - \mathbf{P}_{\mathbf{S}}$ be the projection onto the complement of the row span of $\mathbf{S}^T \mathbf{B}$. By (8):

$$\bar{\mathbf{P}}_{\mathbf{S}} \mathbf{B}^T \mathbf{B} \bar{\mathbf{P}}_{\mathbf{S}} \preceq C \cdot \bar{\mathbf{P}}_{\mathbf{S}} \mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} \bar{\mathbf{P}}_{\mathbf{S}} + \lambda \bar{\mathbf{P}}_{\mathbf{S}} \mathbf{I} \bar{\mathbf{P}}_{\mathbf{S}}. \quad (9)$$

Since $\bar{\mathbf{P}}_{\mathbf{S}}$ projects to the complement of the row span of $\mathbf{S}^T \mathbf{B}$, $\mathbf{S}^T \mathbf{B} \bar{\mathbf{P}}_{\mathbf{S}} = \mathbf{0}$. So (9) gives:

$$\bar{\mathbf{P}}_{\mathbf{S}} \mathbf{B}^T \mathbf{B} \bar{\mathbf{P}}_{\mathbf{S}} \preceq \mathbf{0} + \lambda \bar{\mathbf{P}}_{\mathbf{S}} \mathbf{I} \bar{\mathbf{P}}_{\mathbf{S}} \preceq \lambda \mathbf{I}.$$

In other notation, $\|\bar{\mathbf{P}}_{\mathbf{S}} \mathbf{B}^T \mathbf{B} \bar{\mathbf{P}}_{\mathbf{S}}\|_2 \leq \lambda$. This in turn implies $\|\mathbf{B} \bar{\mathbf{P}}_{\mathbf{S}} \mathbf{B}^T\|_2 \leq \lambda$ and hence:

$$\mathbf{B} \bar{\mathbf{P}}_{\mathbf{S}} \mathbf{B}^T = \mathbf{B} (\mathbf{I} - \mathbf{P}_{\mathbf{S}}) \mathbf{B}^T \preceq \lambda \mathbf{I}.$$

Rearranging and using $\mathbf{K} = \mathbf{B} \mathbf{B}^T$ and $\tilde{\mathbf{K}} = \mathbf{B} \mathbf{P}_{\mathbf{S}} \mathbf{B}^T$ gives the result. A Chernoff bound (see Lemma 11, Appendix A), gives that with probability $1 - \delta/2$, $s \leq 2 \sum_i p_i$, completing the theorem. \square

Often a regularization parameter λ is specified for a learning task, and for near optimal performance on this task, we set the approximation factor in Theorem 3 to $\epsilon \lambda$. In this case we have:

Corollary 4 (Tighter spectral error kernel approximation). *If we use RLS-Nyström to sample $O\left(\frac{d_{\text{eff}}^\lambda}{\epsilon} \log \frac{d_{\text{eff}}^\lambda}{\delta \epsilon}\right)$ landmarks by the $\epsilon \lambda$ -ridge leverage scores then, with probability $1 - \delta$, $\tilde{\mathbf{K}}$ satisfies*

$$\tilde{\mathbf{K}} \preceq \mathbf{K} \preceq \tilde{\mathbf{K}} + \epsilon \lambda \mathbf{I}. \quad (10)$$

Proof. This useful statement follows directly from Theorem 3 by simply observing that $d_{\text{eff}}^{\epsilon \lambda} \leq d_{\text{eff}}^\lambda / \epsilon$, by the fact that $(\mathbf{K} + \epsilon \lambda \mathbf{I})^{-1} \preceq \frac{1}{\epsilon} (\mathbf{K} + \lambda \mathbf{I})^{-1}$. \square

Corollary 4 is sufficient to prove that $\tilde{\mathbf{K}}$ can be used in place of \mathbf{K} without sacrificing performance on kernel ridge regression and canonical correlation tasks (see [AM15] and [Wan16]). We also use it to prove a *projection-cost preservation* guarantee (Theorem 14, Appendix B). Specifically, we show that if $O((k \log k)/\epsilon)$ landmarks are sampled with an appropriately chosen ridge parameter λ , then for any rank- k projection matrix \mathbf{X} , $\tilde{\mathbf{K}}$ will satisfy, for some fixed $c > 0$:

$$\text{tr}(\mathbf{K} - \mathbf{X} \mathbf{K} \mathbf{X}) \leq \text{tr}(\tilde{\mathbf{K}} - \mathbf{X} \tilde{\mathbf{K}} \mathbf{X}) + c \leq (1 + \epsilon) \text{tr}(\mathbf{K} - \mathbf{X} \mathbf{K} \mathbf{X}). \quad (11)$$

(11) allows us to prove approximation guarantees for kernel PCA and k -means clustering. Projection-cost preservation has proven a powerful concept in the matrix sketching literature [FSS13, CEM⁺15, CMM17, BWZ16, CW17]. We hope that an explicit guarantee for kernels will lead to applications of RLS-Nyström beyond those considered in this work.

Our results on downstream learning bounds that can be derived from Theorem 3 are summarized in Table 1. Details can be found in Appendices B and C.

Application	Downstream Guarantee	Relevant Theorem	Space to store $\tilde{\mathbf{K}}$	Time to compute $\tilde{\mathbf{K}}$
Kernel Ridge Regression w/ Parameter λ	$(1 + \epsilon)$ relative error risk bound	Thm 15	$\tilde{O}(\frac{nd_{\text{eff}}^\lambda}{\epsilon})$	$\tilde{O}(\frac{n(d_{\text{eff}}^\lambda)^2}{\epsilon^2}) + \tilde{O}(\frac{nd_{\text{eff}}^\lambda}{\epsilon})$ kernel evals.
Kernel k -means Clustering	$(1 + \epsilon)$ relative error	Thm 16	$\tilde{O}(\frac{nk}{\epsilon})$	$\tilde{O}(\frac{nk^2}{\epsilon^2}) + \tilde{O}(\frac{nk}{\epsilon})$ kernel evals.
Rank k Kernel PCA	$(1 + \epsilon)$ relative Frobenius norm error	Thm 17	$\tilde{O}(\frac{nk}{\epsilon})$	$\tilde{O}(\frac{nk^2}{\epsilon^2}) + \tilde{O}(\frac{nk}{\epsilon})$ kernel evals.
Kernel CCA w/ Regularization Params λ_x, λ_y	ϵ additive error to canonical correlation	Thm 18	$\tilde{O}(\frac{nd_{\text{eff}}^{\lambda_x} + nd_{\text{eff}}^{\lambda_y}}{\epsilon})$	$\tilde{O}(\frac{n(d_{\text{eff}}^{\lambda_x})^2 + n(d_{\text{eff}}^{\lambda_y})^2}{\epsilon^2}) + \tilde{O}(\frac{nd_{\text{eff}}^{\lambda_x} + nd_{\text{eff}}^{\lambda_y}}{\epsilon})$ kernel evals.

* For conciseness, $\tilde{O}(\cdot)$ hides log factors in the failure probability, d_{eff} , and k .

Table 1: Downstream guarantees for $\tilde{\mathbf{K}}$ obtained from RLS-Nyström (Algorithm 1). For all problems, the runtime and space cost depends linearly on the number of training data points n .

4 Recursive sampling for efficient RLS-Nyström

Having established strong approximation guarantees for RLS-Nyström, it remains to provide an efficient implementation. Specifically, Step 1 of Algorithm 1 naively requires $\Theta(n^3)$ time. We show that significant acceleration is possible using a recursive sampling approach, which is adapted from techniques developed in [CLM⁺15] and [CMM17].

4.1 Ridge leverage score approximation via uniform sampling

The key idea is to approximate the ridge leverage scores of \mathbf{K} using a uniform sample of the data points. To ensure accuracy, the sample must be large – consisting of $1/2$ of the points. We later show how to recursively approximate this large sample to achieve our final runtimes. We first prove:

Lemma 5. *For any $\mathbf{B} \in \mathbb{R}^{n \times n}$ with $\mathbf{B}\mathbf{B}^T = \mathbf{K}$ and $\mathbf{S} \in \mathbb{R}^{n \times s}$ chosen by sampling each data point independently with probability $1/2$, let*

$$\tilde{l}_i^\lambda = \mathbf{b}_i^T (\mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda \mathbf{I})^{-1} \mathbf{b}_i \quad (12)$$

and $p_i = \min\{1, 16\tilde{l}_i^\lambda \log(\sum_i \tilde{l}_i^\lambda / \delta)\}$ for any $\delta \in (0, 1/8)$. Then with probability at least $1 - \delta$:

1. $\tilde{l}_i^\lambda \geq l_i^\lambda$ for all i .
2. $\sum_i p_i \leq 64 \sum_i l_i^\lambda \log(\sum_i l_i^\lambda / \delta)$.

The first condition ensures that the approximate scores \tilde{l}_i^λ suffice for use in Algorithm 1. The second ensures that the Nyström approximation obtained will have, up to constant factors, the same size as if we used the true ridge leverage scores. Note that it is not obvious how to compute \tilde{l}_i^λ using the formula in (12) without explicitly forming \mathbf{B} . We discuss how to do this in Section 4.2.

Proof. The first bound follows trivially since $\mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} \preceq \mathbf{B}^T \mathbf{B}$ so:

$$\tilde{l}_i^\lambda = \mathbf{b}_i^T (\mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda \mathbf{I})^{-1} \mathbf{b}_i \geq \mathbf{b}_i^T (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1} \mathbf{b}_i = l_i^\lambda.$$

The challenge is showing the second bound. The key observation is that there exists a diagonal reweighting matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$, $\mathbf{0} \preceq \mathbf{W} \preceq \mathbf{I}$ such that for all i , $l_i^\lambda(\mathbf{W}\mathbf{K}\mathbf{W}) \leq \alpha$ where $\alpha \stackrel{\text{def}}{=} \frac{1}{2} \cdot \frac{1}{16 \log(\sum l_i^\lambda / \delta)}$. This bound ensures that uniformly sampling rows with probability $1/2$ from the *reweighted kernel* $\mathbf{W}\mathbf{K}\mathbf{W}$ is a valid ridge leverage score sampling. Additionally, $|\{i : \mathbf{W}_{i,i} < 1\}| \leq 32 \log(\sum l_i^\lambda / \delta) \cdot \sum l_i^\lambda$. That is, we do not need to reweight too many columns to achieve the ridge leverage score upper bound.

Although \mathbf{W} is never actually computed, its existence can be proved algorithmically: we can construct a valid \mathbf{W} by iteratively considering any i with $l_i^\lambda(\mathbf{W}\mathbf{K}\mathbf{W}) \geq \alpha$. Since $\lambda > 0$, it is always possible to decrease the ridge leverage score to exactly α by decreasing $\mathbf{W}_{i,i}$ sufficiently.

It is clear from the interpretation of Definition 1 given in (4) that decreasing $\mathbf{W}_{i,i}$, which corresponds to decreasing the weight of one row of \mathbf{B} , will only increase the ridge leverage scores of other rows. So, any reweighted row will always maintain leverage score $\geq \alpha$ as other rows are reweighted. Theorem 2 of [CLM⁺15] demonstrates rigorously that the leverage scores of these reweighted rows in fact converge to α . Furthermore, since $\mathbf{W} \preceq \mathbf{I}$, it is not hard to show (see Lemma 19 in Appendix D.1):

$$\sum_i l_i^\lambda(\mathbf{W}\mathbf{K}\mathbf{W}) \leq \sum_i l_i^\lambda(\mathbf{K}) \stackrel{\text{def}}{=} \sum_i l_i^\lambda.$$

Thus, since each reweighted row has $l_i^\lambda(\mathbf{W}\mathbf{K}\mathbf{W}) \geq \alpha$, $\alpha \cdot |\{i : \mathbf{W}_{i,i} < 1\}| \leq \sum_i l_i^\lambda$ and so:

$$|\{i : \mathbf{W}_{i,i} < 1\}| \leq \frac{1}{\alpha} \sum_i l_i^\lambda = 32 \log\left(\sum l_i^\lambda / \delta\right) \cdot \sum l_i^\lambda.$$

We can now bound $\sum_i p_i$. For any i that is reweighted by \mathbf{W} we just trivially bound $p_i \leq 1$. Since $l_i^\lambda(\mathbf{W}\mathbf{K}\mathbf{W}) \leq \frac{1}{2} \cdot \frac{1}{16 \log(\sum l_i^\lambda / \delta)}$ for all i , and since \mathbf{S} samples each i with probability $1/2$, by the matrix Bernstein bound of Lemma 11, with probability $1 - \delta/2$:

$$\frac{1}{2}(\mathbf{B}^T \mathbf{W}^2 \mathbf{B} + \lambda \mathbf{I}) \preceq (\mathbf{B}^T \mathbf{W} \mathbf{S} \mathbf{S}^T \mathbf{W} \mathbf{B} + \lambda \mathbf{I}) \preceq \frac{3}{2}(\mathbf{B}^T \mathbf{W}^2 \mathbf{B} + \lambda \mathbf{I}).$$

Hence:

$$\begin{aligned} \tilde{l}_i^\lambda &= \mathbf{b}_i^T (\mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda \mathbf{I})^{-1} \mathbf{b}_i \leq \mathbf{b}_i^T (\mathbf{B}^T \mathbf{W} \mathbf{S} \mathbf{S}^T \mathbf{W} \mathbf{B} + \lambda \mathbf{I})^{-1} \mathbf{b}_i \\ &\leq 2 \mathbf{b}_i^T (\mathbf{B}^T \mathbf{W}^2 \mathbf{B} + \lambda \mathbf{I})^{-1} \mathbf{b}_i \\ &= 2 l_i^\lambda(\mathbf{W} \mathbf{B} \mathbf{B}^T \mathbf{W}) = 2 l_i^\lambda(\mathbf{W} \mathbf{K} \mathbf{W}). \end{aligned}$$

Again using that $\mathbf{W} \preceq \mathbf{I}$ and Lemma 19, $\sum_{\{i: \mathbf{W}_{i,i}=1\}} \tilde{l}_i^\lambda \leq 2 \sum_i l_i^\lambda$. Overall:

$$\begin{aligned} \sum_i p_i &= \sum_{\{i: \mathbf{W}_{i,i} < 1\}} p_i + \sum_{\{i: \mathbf{W}_{i,i}=1\}} p_i \\ &\leq |\{i : \mathbf{W}_{i,i} < 1\}| + 32 \log\left(\sum l_i^\lambda / \delta\right) \cdot \sum_i l_i^\lambda \\ &= 64 \log\left(\sum l_i^\lambda / \delta\right) \cdot \sum_i l_i^\lambda. \end{aligned}$$

□

4.2 Computing ridge leverage scores from a sample

In order to utilize Lemma 5 we must show how to efficiently compute \tilde{l}_i^λ via formula (12) *without explicitly forming* either \mathbf{K} or \mathbf{B} . We prove the following:

Lemma 6. *For any sampling matrix $\mathbf{S} \in \mathbb{R}^{n \times s}$, and any $\lambda > 0$:*

$$\tilde{l}_i^\lambda \stackrel{\text{def}}{=} \mathbf{b}_i^T (\mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda \mathbf{I})^{-1} \mathbf{b}_i = \frac{1}{\lambda} \left(\mathbf{K} - \mathbf{K} \mathbf{S} (\mathbf{S}^T \mathbf{K} \mathbf{S} + \lambda \mathbf{I})^{-1} \mathbf{S}^T \mathbf{K} \right)_{i,i}.$$

It follows that we can compute \tilde{l}_i^λ for all i in $O(ns^2)$ time using just $O(ns)$ kernel evaluations.

Proof. Using the SVD write $\mathbf{S}^T \mathbf{B} = \bar{\mathbf{U}} \bar{\Sigma} \bar{\mathbf{V}}^T$. $\bar{\mathbf{V}} \in \mathbb{R}^{n \times s}$ forms an orthonormal basis for the row span of $\mathbf{S}^T \mathbf{B}$. Let $\bar{\mathbf{V}}_\perp$ be span for the nullspace of $\mathbf{S}^T \mathbf{B}$. Then we can rewrite \tilde{l}_i^λ as:

$$\tilde{l}_i^\lambda = \mathbf{b}_i^T (\mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda \mathbf{I})^{-1} \mathbf{b}_i = \mathbf{b}_i^T [\bar{\mathbf{V}}, \bar{\mathbf{V}}_\perp] (\bar{\Sigma}^2 + \lambda \mathbf{I})^{-1} [\bar{\mathbf{V}}, \bar{\mathbf{V}}_\perp]^T \mathbf{b}_i.$$

Here we're abusing notation a bit by letting $\bar{\Sigma}$ represent an $n \times n$ diagonal matrix whose first s entries are the singular values of $\mathbf{S}^T \mathbf{B}$ and whose remaining entries are all equal to 0. Now:

$$\tilde{l}_i^\lambda = \mathbf{b}_i^T [\bar{\mathbf{V}}, \bar{\mathbf{V}}_\perp] (\bar{\Sigma}^2 + \lambda \mathbf{I})^{-1} [\bar{\mathbf{V}}, \bar{\mathbf{V}}_\perp]^T \mathbf{b}_i = \frac{1}{\lambda} \mathbf{b}_i^T \bar{\mathbf{V}}_\perp^T \bar{\mathbf{V}}_\perp \mathbf{b}_i + \mathbf{b}_i^T \bar{\mathbf{V}} (\bar{\Sigma}^2 + \lambda \mathbf{I})^{-1} \bar{\mathbf{V}}^T \mathbf{b}_i. \quad (13)$$

Focusing on the second term of (13),

$$\begin{aligned} \mathbf{b}_i^T \bar{\mathbf{V}} (\bar{\Sigma}^2 + \lambda \mathbf{I})^{-1} \bar{\mathbf{V}}^T \mathbf{b}_i &= \mathbf{b}_i^T \bar{\mathbf{V}} \frac{1}{\lambda} (\mathbf{I} - \bar{\Sigma}^2 (\bar{\Sigma}^2 + \lambda \mathbf{I})^{-1}) \bar{\mathbf{V}}^T \mathbf{b}_i \\ &= \frac{1}{\lambda} \mathbf{b}_i^T \bar{\mathbf{V}} \bar{\mathbf{V}}^T \mathbf{b}_i - \frac{1}{\lambda} \mathbf{b}_i^T \bar{\mathbf{V}} (\bar{\Sigma}^2 (\bar{\Sigma}^2 + \lambda \mathbf{I})^{-1}) \bar{\mathbf{V}}^T \mathbf{b}_i. \end{aligned} \quad (14)$$

Focusing on the second term of (14),

$$\begin{aligned} \mathbf{b}_i^T \bar{\mathbf{V}} (\bar{\Sigma}^2 (\bar{\Sigma}^2 + \lambda \mathbf{I})^{-1}) \bar{\mathbf{V}}^T \mathbf{b}_i &= \mathbf{b}_i^T \bar{\mathbf{V}} \bar{\Sigma} \bar{\mathbf{U}}^T \bar{\mathbf{U}} (\bar{\Sigma}^2 + \lambda \mathbf{I})^{-1} \bar{\mathbf{U}}^T \bar{\mathbf{U}} \bar{\Sigma} \bar{\mathbf{V}}^T \mathbf{b}_i \\ &= \mathbf{b}_i^T \mathbf{B}^T \mathbf{S} (\mathbf{S}^T \mathbf{K} \mathbf{S} + \lambda \mathbf{I})^{-1} \mathbf{S}^T \mathbf{B} \mathbf{b}_i. \end{aligned}$$

Substituting back into (14) and then (13), we conclude that:

$$\begin{aligned} \tilde{l}_i^\lambda &= \frac{1}{\lambda} \mathbf{b}_i^T \bar{\mathbf{V}}_\perp^T \bar{\mathbf{V}}_\perp \mathbf{b}_i + \frac{1}{\lambda} \mathbf{b}_i^T \bar{\mathbf{V}} \bar{\mathbf{V}}^T \mathbf{b}_i - \frac{1}{\lambda} \mathbf{b}_i^T \mathbf{B}^T \mathbf{S} (\mathbf{S}^T \mathbf{K} \mathbf{S} + \lambda \mathbf{I})^{-1} \mathbf{S}^T \mathbf{B} \mathbf{b}_i \\ &= \frac{1}{\lambda} \mathbf{b}_i^T \mathbf{b}_i - \frac{1}{\lambda} \mathbf{b}_i^T \mathbf{B}^T \mathbf{S} (\mathbf{S}^T \mathbf{K} \mathbf{S} + \lambda \mathbf{I})^{-1} \mathbf{S}^T \mathbf{B} \mathbf{b}_i \\ &= \frac{1}{\lambda} \mathbf{K}_{i,i} - \frac{1}{\lambda} \left(\mathbf{K} \mathbf{S} (\mathbf{S}^T \mathbf{K} \mathbf{S} + \lambda \mathbf{I})^{-1} \mathbf{S}^T \mathbf{K} \right)_{i,i}. \end{aligned}$$

We can compute $(\mathbf{S}^T \mathbf{K} \mathbf{S} + \lambda \mathbf{I})^{-1}$ in $O(s^3) \leq O(ns^2)$ time and $O(s^2) \leq O(ns)$ kernel evaluations. Given this inverse, computing the diagonal entries of $\mathbf{K} \mathbf{S} (\mathbf{S}^T \mathbf{K} \mathbf{S} + \lambda \mathbf{I})^{-1} \mathbf{S}^T \mathbf{K}$ requires just $O(ns)$ kernel evaluations to form $\mathbf{K} \mathbf{S}$ and $O(ns^2)$ time to perform the necessary multiplications. Finally, computing the diagonal entries of \mathbf{K} requires n additional kernel evaluations. \square

4.3 Recursive RLS-Nyström

We are finally ready to use Lemmas 5 and 6 to give an efficient recursive method for ridge leverage score Nyström approximation.

Algorithm 2 RECURSIVE RLS-NYSTRÖM SAMPLING.

input: $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$, kernel function $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, ridge parameter λ , $\delta \in (0, 1/8)$

output: weighted sampling matrix $\mathbf{S} \in \mathbb{R}^{n \times s}$

- 1: Choose \mathbf{S}_0 by sampling each data point independently with probability $1/2$.
 - 2: If \mathbf{S}_0 has > 16 columns, apply Algorithm 2 **recursively** to $\mathbf{S}_0^T \mathbf{K} \mathbf{S}_0$ with $\delta \leftarrow \delta/2$ to compute \mathbf{S}_1 . Else set $\mathbf{S}_1 = \mathbf{S}_0$.
 - 3: $\tilde{l}_i^\lambda := \frac{3}{2\lambda} \left(\mathbf{K} - \mathbf{K} \mathbf{S}_1 (\mathbf{S}_1^T \mathbf{K} \mathbf{S}_1 + \lambda \mathbf{I})^{-1} \mathbf{S}_1^T \mathbf{K} \right)_{i,i}$ ▷ Equals $\frac{3}{2}(\mathbf{B}(\mathbf{B}^T \mathbf{S}_1 \mathbf{S}_1^T \mathbf{B} + \lambda \mathbf{I})^{-1} \mathbf{B}^T)_{i,i}$
 - 4: $p_i := \min\{1, \tilde{l}_i^\lambda \cdot 16 \log(\sum \tilde{l}_i^\lambda / \delta)\}$
 - 5: **return** \mathbf{S} chosen by sampling i with probability p_i and reweighting selected columns by $1/\sqrt{p_i}$.
-

We show that the output of Algorithm 2, \mathbf{S} , is sampled according to approximate ridge leverage scores for \mathbf{K} and thus satisfies the approximation guarantee of Theorem 3.

Theorem 7 (Main Result). *Let $\mathbf{S} \in \mathbb{R}^{n \times s}$ be computed by Algorithm 2. With probability $1 - 2\delta$, $s \leq 384 \cdot d_{\text{eff}}^\lambda \log(d_{\text{eff}}^\lambda / \delta)$, \mathbf{S} is sampled by overestimates of the λ -ridge leverage scores of \mathbf{K} , and the Nyström approximation $\tilde{\mathbf{K}} = \mathbf{K} \mathbf{S} (\mathbf{S}^T \mathbf{K} \mathbf{S})^+ \mathbf{S}^T \mathbf{K}$ satisfies the approximation guarantee of Theorem 3. Algorithm 2 uses $O(ns)$ kernel evaluations and $O(ns^2)$ computation time.*

Note that in Algorithm 2 the columns of \mathbf{S} are reweighted by $1/\sqrt{p_i}$. This is necessary for the recursive calls, but not for the final Nyström approximation, which is unaffected by column weights.

Theorem 7 follows from the recursive invariant:

Theorem 8. *With probability $1 - 2\delta$, Algorithm 2 returns \mathbf{S} satisfying, for any \mathbf{B} with $\mathbf{B} \mathbf{B}^T = \mathbf{K}$:*

$$\frac{1}{2}(\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I}) \preceq (\mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda \mathbf{I}) \preceq \frac{3}{2}(\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I}). \quad (15)$$

\mathbf{S} has $s \leq 384 \cdot d_{\text{eff}}^\lambda \log(d_{\text{eff}}^\lambda / \delta)$ columns and the algorithm performs $O(ns)$ kernel evaluations and runs in $O(ns^2)$ time.

Proof. Assume by induction that after forming \mathbf{S}_0 via uniformly sampling, the recursive call to Algorithm 2 returns \mathbf{S}_1 satisfying:

$$\frac{1}{2}(\mathbf{B}^T \mathbf{S}_0 \mathbf{S}_0^T \mathbf{B} + \lambda \mathbf{I}) \preceq (\mathbf{B}^T \mathbf{S}_1 \mathbf{S}_1^T \mathbf{B} + \lambda \mathbf{I}) \preceq \frac{3}{2}(\mathbf{B}^T \mathbf{S}_0 \mathbf{S}_0^T \mathbf{B} + \lambda \mathbf{I}).$$

Then for all i , \tilde{l}_i^λ is greater than the approximate leverage scores we would have obtained if we had computed them based on \mathbf{S}_0 (note the $3/2$ factor over-sampling in Step 3). So if we sample by \tilde{l}_i^λ , by Lemmas 5 and 11, with probability $1 - \delta$, \mathbf{S} will satisfy (15) and have $1/2 \cdot \sum p_i \leq s \leq 2 \sum p_i$ columns where: $\sum p_i \leq 2 \cdot 3/2 \cdot 64 \sum_i \tilde{l}_i^\lambda \log(\sum_i \tilde{l}_i^\lambda / \delta) \leq 192 \cdot d_{\text{eff}}^\lambda \log(d_{\text{eff}}^\lambda / \delta)$. By a union bound, since the recursive call is run with failure probability $\delta/2$, the overall failure probability of Algorithm 2 is $\leq \delta + 2 \cdot (\delta/2) = 2\delta$.

To bound runtime, let n_j be the number of points passed to the j^{th} recursive call of Algorithm 2 and let s_j be the number of points sampled by \mathbf{S}_1 at that level. Applying Lemma 6, the total cost to compute \mathbf{S} at level j , excluding the recursive computation of \mathbf{S}_1 , is $O(n_j s_j)$ kernel evaluations and $O(n_j s_j^2 + s_j^3) = O(n_j s_j^2)$ additional time. By induction, and applying Lemma 19 to show that the statistical dimension of $\mathbf{S}_0^T \mathbf{K} \mathbf{S}_0$ is less than that of \mathbf{K} , we have:

$$s_j \leq 384 \log \left(\frac{\sum_i l_i^\lambda (\mathbf{S}_0^T \mathbf{K} \mathbf{S}_0)}{\delta/2^j} \right) \sum_i l_i^\lambda (\mathbf{S}_0^T \mathbf{K} \mathbf{S}_0) \leq 384 d_{\text{eff}}^\lambda \cdot \log \left(\frac{d_{\text{eff}}^\lambda}{\delta/2^j} \right) = O(j \cdot s).$$

Since $n_j = O(n/2^j)$ in expectation and with high probability, our total runtime is $O(ns^2 \cdot (1 + 1/2 + 2^2/4 + 3^2/8 + \dots + \log^2 n/n)) = O(ns^2)$ by the fact that $\sum_{j=1}^\infty \frac{j^2}{2^j} = 6$. \square

Proof of Theorem 7. The theorem follows immediately since Theorem 8 guarantees that in the last level of recursion \mathbf{K} is sampled by over approximations of the ridge leverage scores. The runtime follows from Theorem 8 and the fact that it is possible to compute $\mathbf{K} \mathbf{S}$ using $O(ns)$ kernel evaluations and $(\mathbf{S}^T \mathbf{K} \mathbf{S})^+$ using $O(ns^2 + s^3) = O(ns^2)$ additional time. \square

5 Empirical Evaluation

We conclude with an empirical evaluation of our recursive Nyström method. We first introduce a variant of Algorithm 2 where, instead of choosing a regularization parameter λ , the user sets a sample size s and λ is automatically determined such that $s = \Theta(d_{\text{eff}}^\lambda \cdot \log(d_{\text{eff}}^\lambda/\delta))$. This variant is practically appealing as it essentially yields the best possible approximation to \mathbf{K} for a fixed sample budget. Additionally, it is necessary in applications to kernel rank- k PCA and k -means clustering, when λ is unknown, but where we set $s = \Theta(k \log k)$ (see Appendices B and C).

5.1 Recursive RLS-Nyström algorithm for fixed sample size

Given a fixed sample size s , we will control λ using the following fact:

Fact 9 (Proven in (25), Appendix B). *For any \mathbf{K} and integer k , for $\lambda = \frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})$, $d_{\text{eff}}^\lambda \leq 2k$.*

If we choose k such that $s = \Theta(k \log k)$ then setting λ as above will yield an RLS-Nyström approximation with approximately s sampled columns. The details are given in Algorithm 3.

Algorithm 3 RECURSIVE RLS-NYSTRÖM SAMPLING, FIXED SAMPLE SIZE.

input: $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$, kernel function $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, sample size s , $\delta \in (0, 1/8)$

output: sampling matrix $\mathbf{S} \in \mathbb{R}^{n \times s'}$.

- 1: Choose \mathbf{S}_0 by sampling each data point independently with probability $1/2$.
 - 2: If \mathbf{S}_0 has > 16 columns, apply Algorithm 3 **recursively** to $\mathbf{S}_0^T \mathbf{K} \mathbf{S}_0$ with $\delta \leftarrow \delta/2$ to compute \mathbf{S}_1 . Else set $\mathbf{S}_1 = \mathbf{S}_0$.
 - 3: Set k to the maximum integer such that $s \geq 640k \log(2k/\delta)$.
 - 4: $\tilde{\lambda} := \frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{S}_1^T \mathbf{K} \mathbf{S}_1)$ \triangleright Approximate λ
 - 5: $\tilde{l}_i^\lambda := \frac{5}{\lambda} \left(\mathbf{K} - \mathbf{K} \mathbf{S}_1 \left(\mathbf{S}_1^T \mathbf{K} \mathbf{S}_1 + \tilde{\lambda} \mathbf{I} \right)^{-1} \mathbf{S}_1^T \mathbf{K} \right)_{i,i}$ \triangleright Equals $(\mathbf{B}(\mathbf{B}^T \mathbf{S}_1 \mathbf{S}_1^T \mathbf{B} + \tilde{\lambda} \mathbf{I})^{-1} \mathbf{B}^T)_{i,i}$
 - 6: $p_i := \min\{1, \tilde{l}_i^\lambda \cdot 16 \log(2k/\delta)\}$
 - 7: **return** \mathbf{S} chosen by sampling i with probability p_i and reweighting selected columns by $1/\sqrt{p_i}$.
-

Theorem 10. Let k be any integer with $s \geq 640k \log(2k/\delta)$ and $\lambda = \frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})$. Let $\mathbf{S} \in \mathbb{R}^{n \times s'}$ be computed by Algorithm 3. With probability $1 - 2\delta$, $s' \leq 2s$, \mathbf{S} is sampled by overestimates of the λ -ridge leverage scores of \mathbf{K} , and the Nyström approximation $\tilde{\mathbf{K}} = \mathbf{K}\mathbf{S}(\mathbf{S}^T\mathbf{K}\mathbf{S})^+ \mathbf{S}^T\mathbf{K}$ satisfies the guarantee of Theorem 3. Algorithm 3 uses $O(ns)$ kernel evaluations and $O(ns^2)$ runtime.

For the λ given in Theorem 10, we have $d_{\text{eff}}^\lambda = \Theta(k)$. Hence, since we set $s = \Theta(k \log k)$, additive error λ is essentially the smallest we can obtain using an s sample Nyström approximation. The proof of Theorem 10 is similar to that of Theorem 7. We defer it to Appendix D.

5.2 Performance of Recursive RLS-Nyström for kernel approximation

We evaluate Algorithm 3 on the datasets listed in Table 2, comparing against the classic Nyström method with uniform sampling [WS01] and the random Fourier features method [RR07]. Implementations were in MATLAB and run on a 2.6 GHz Intel Core i7 with 16GB of memory.

Dataset	# of Data Points n	# of Features d	Link
YearPredictionMSD	515345	90	https://archive.ics.uci.edu/ml/datasets/YearPredictionMSD
Covertypes	581012	54	https://archive.ics.uci.edu/ml/datasets/Covertypes
Cod-RNA	331152	8	https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/
Adult	48842	110	https://archive.ics.uci.edu/ml/datasets/Adult

Table 2: Datasets downloaded from UCI ML Repository [Lic13], except Cod-RNA [UKM06].

For each dataset, we split categorical features into binary indicatory features and mean center and normalize all features to have variance 1. We use a Gaussian kernel for all tests, with the width parameter σ selected via cross validation on regression and classification tasks. To compute $\|\mathbf{K} - \tilde{\mathbf{K}}\|_2$, we only process a random subset of 20k data points since otherwise multiplying by the full kernel matrix \mathbf{K} to compute $\|\mathbf{K} - \tilde{\mathbf{K}}\|_2$ is prohibitively expensive. Experiments on the full kernel matrices are discussed in Section 5.3.

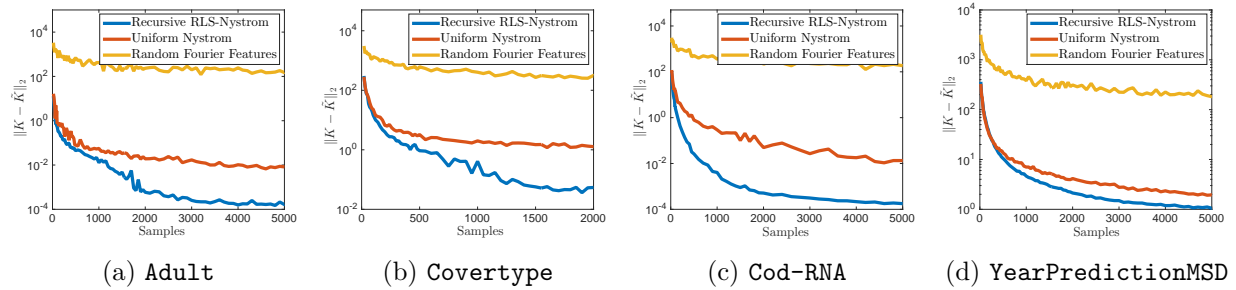


Figure 2: For a given number of samples, Recursive RLS-Nyström yields approximations with lower error, measured by $\|\mathbf{K} - \tilde{\mathbf{K}}\|_2$. Error is plotted on a logarithmic scale, averaged over 10 trials.

Figure 2 confirms that Recursive RLS-Nyström consistently obtains better kernel approximation error than the other methods. The advantage of Nyström over random Fourier features is substantial – this is unsurprising as the Nyström methods are data dependent and based on data projection,

as opposed to pointwise approximation of \mathbf{K} . Even between the Nyström methods there is a substantial difference in kernel approximation, especially for large sample sizes.

As we can see in Figure 3, with the exception of **YearPredictionMSD**, the better quality of the landmarks obtained with Recursive RLS-Nyström translates into runtime improvements. While the cost *per sample* is higher for our method at $O(nd + ns)$ time versus $O(nd + s^2)$ for uniform Nyström and $O(nd)$ for random Fourier features, since RLS-Nyström requires fewer samples it more quickly obtains $\tilde{\mathbf{K}}$ with a given accuracy. $\tilde{\mathbf{K}}$ will also have lower rank, which can accelerate processing in downstream applications. For example, to achieve $\|\mathbf{K} - \tilde{\mathbf{K}}\|_2 \leq 1$ for the **Covertypes** dataset, Recursive RLS-Nyström requires 650 samples in comparison to 3800 for uniform Nyström.

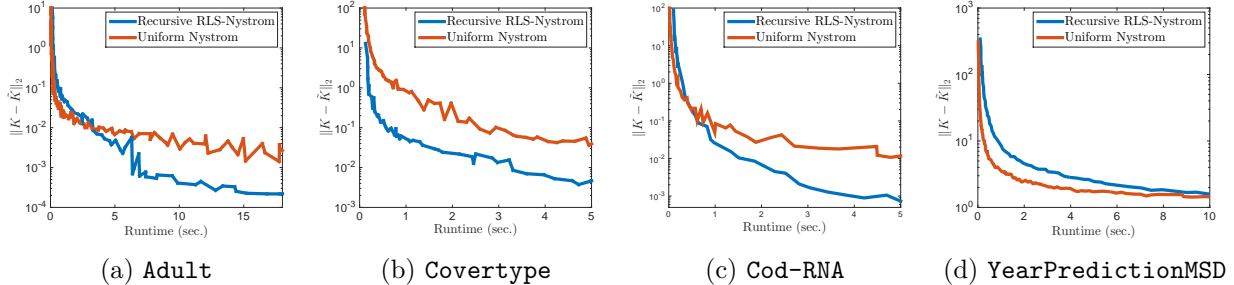


Figure 3: Especially for small error, Recursive RLS-Nyström typically obtains a fixed level of approximation faster than uniform sampling. It only underperformed uniform sampling for the **YearPredictionMSD** dataset. Results for random Fourier features are excluded from this plot: while the method is faster than Nyström, it never obtained high enough accuracy to be directly comparable. Error is plotted on a log scale, with results averaged over 10 trials.

5.2.1 Accelerated recursive method

While Recursive RLS-Nyström typically outperforms classic Nyström, on datasets with relatively uniform ridge leverage scores, such as **YearPredictionMSD**, it only narrowly beats uniform sampling in terms accuracy. As a result it incurs a higher runtime cost since it is slower per sample.

To combat this issue we implement a simple heuristic modification of our algorithm. We note that the final cost of computing the Nyström factors \mathbf{KS} and $(\mathbf{S}^T \mathbf{KS})^+$ is $O(ns + s^3)$ for both methods. Recursive RLS-Nyström is only slower because computing leverage scores at intermediate levels of recursion takes $O(ns^2)$ time (Step 5, Algorithm 3). This cost can be improved by simply adjusting the regularization λ to restrict the sample size on each recursive call to be $< s$. Specifically, we can balance runtimes by taking $\approx \sqrt{(ns + s^3)/n}$ samples on lower levels.

Doing so improves our runtime, bringing the per sample cost down to approximately that of random Fourier features and uniform Nyström (Figure 4a) while nearly maintaining the same approximation quality. For datasets such as **Covertypes** in which Recursive RLS-Nyström performs significantly better than uniform sampling, so does the accelerated method (see Figure 4b). However, the performance of the accelerated method does not degrade when leverage scores are relatively uniform – it still offers the best runtime to approximation quality tradeoff (Figure 4c).

We note that further runtime improvements may be possible. Subsequent work extends fast ridge leverage score methods to distributed and streaming environments [CLV17]. Empirical evaluation of these techniques could lead to even more scalable, high accuracy Nyström methods.

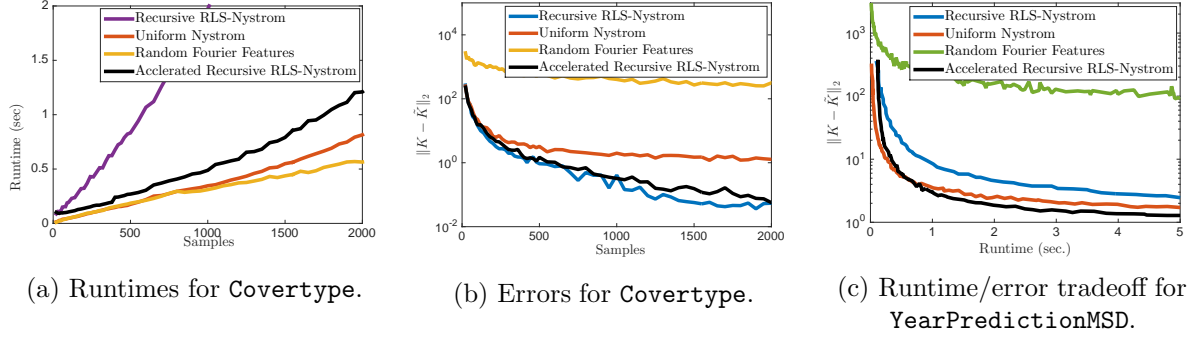


Figure 4: Our accelerated Recursive RLS-Nyström, which undersamples at intermediate recursive calls, nearly matches the *per sample runtime* of random Fourier features and uniform Nyström while still providing approximation nearly as good as the standard Recursive RLS-Nyström. For datasets like **YearPredictionMSD** with relatively uniform kernel leverage scores, the accelerated version offers the best runtime vs. approximation tradeoff. All results are averaged over 10 trials.

5.3 Performance of Recursive RLS-Nyström for learning tasks

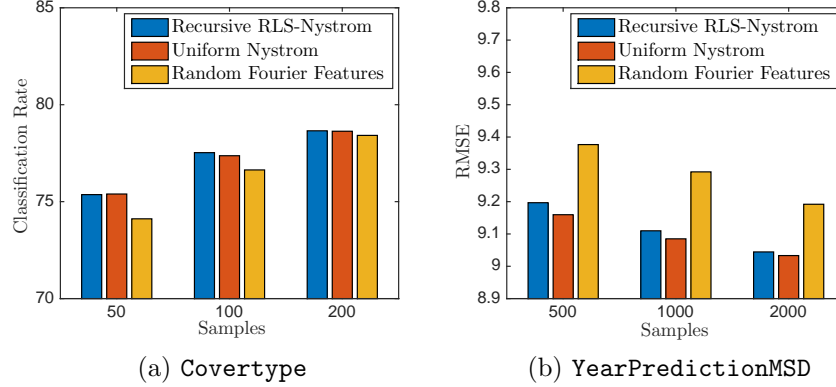


Figure 5: Performance of kernel approximation methods for classification and clustering. For **Covertypes**, classification error is measured in separating Class 2 from the remaining classes. For **YearPredictionMSD**, RMSE is for the unnormalized output. Regularization and kernel parameters are obtained via cross validation on training data. Test results are averaged over 10 trials with a fixed test set, as all three algorithms are randomized.

We conclude by verifying the usefulness of our kernel approximations in downstream learning tasks. We focus on **Covertypes** and **YearPredictionMSD**, which each have approximately $n = 500,000$ data points. While full kernel methods do not scale in this regime, Recursive RLS-Nyström does since its runtime depends linearly on n . For example, on **YearPredictionMSD** the method requires 307 sec. (averaged over 5 trials) to build a 2,000 landmark Nyström approximation for 463,716 training points. Ridge regression using the approximate kernel then requires 208 sec. for a total of 515 sec. In comparison, the fastest method, random Fourier features, required 43 sec. to build a rank 2,000 kernel approximation and 222 sec. for regression, for a total time of 265 sec.

For **Coverttype** we performed classification using the LIBLINEAR support vector machine library. For all sample sizes the SVM dominated runtime cost, so Recursive RLS-Nyström was only marginally slower than uniform Nyström and random Fourier features for a fixed sample size.

In terms of classification performance for **Coverttype** and RMSE error for **YearPredictionMSD**, as can be seen in Figure 5, both Nyström methods outperform random features when using the same number of features. However, we do not see much difference between the two Nyström methods. We leave open understanding why the significantly better kernel approximations discussed in Section 5.2 do not necessarily translate to much better learning performance, or whether they would make a larger difference for other problems.

Acknowledgements

We would like to thank Michael Mahoney for bringing the potential of ridge leverage scores to our attention and suggesting their possible approximation via iterative sampling schemes. We would also like to thank Michael Cohen for pointing out (and fixing) an error in our original manuscript and generally for his close collaboration in our work on leverage score sampling algorithms. Finally, thanks to Haim Avron for pointing out an error in our original analysis.

References

- [AM15] Ahmed Alaoui and Michael W Mahoney. Fast randomized kernel ridge regression with statistical guarantees. In *Advances in Neural Information Processing Systems 28 (NIPS)*, pages 775–783, 2015.
- [AMS01] Dimitris Achlioptas, Frank Mcsherry, and Bernhard Schölkopf. Sampling techniques for kernel methods. In *Advances in Neural Information Processing Systems 14 (NIPS)*, 2001.
- [ANW14] Haim Avron, Huy Nguyen, and David Woodruff. Subspace embeddings for the polynomial kernel. In *Advances in Neural Information Processing Systems 27 (NIPS)*, pages 2258–2266, 2014.
- [Bac13] Francis Bach. Sharp analysis of low-rank kernel matrix approximations. In *Proceedings of the 26th Annual Conference on Computational Learning Theory (COLT)*, 2013.
- [BBV06] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. Kernels as features: On kernels, margins, and low-dimensional mappings. *Machine Learning*, 65(1):79–94, 2006.
- [BJ02] Francis Bach and Michael I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3(Jul):1–48, 2002.
- [BMD09] Christos Boutsidis, Michael W. Mahoney, and Petros Drineas. Unsupervised feature selection for the k -means clustering problem. In *Advances in Neural Information Processing Systems 22 (NIPS)*, pages 153–161, 2009.

- [BW09] Mohamed-Ali Belabbas and Patrick J. Wolfe. Spectral methods in machine learning: New strategies for very large datasets. *Proceedings of the National Academy of Sciences of the USA*, 106:369–374, 2009.
- [BWZ16] Christos Boutsidis, David P. Woodruff, and Peilin Zhong. Optimal principal component analysis in distributed and streaming models. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC)*, 2016.
- [CEM⁺15] Michael B. Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC)*, pages 163–172, 2015.
- [CLL⁺15] Shouyuan Chen, Yang Liu, Michael Lyu, Irwin King, and Shengyu Zhang. Fast relative-error approximation algorithm for ridge regression. In *Proceedings of the 31st Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 201–210, 2015.
- [CLM⁺15] Michael B. Cohen, Yin Tat Lee, Cameron Musco, Christopher Musco, Richard Peng, and Aaron Sidford. Uniform sampling for matrix approximation. In *Proceedings of the 6th Conference on Innovations in Theoretical Computer Science (ITCS)*, pages 181–190, 2015.
- [CLV16] Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Analysis of Nyström method with sequential ridge leverage score sampling. In *Proceedings of the 32nd Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 62–71, 2016.
- [CLV17] Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Distributed adaptive sampling for kernel matrix approximation. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.
- [CMM17] Michael B. Cohen, Cameron Musco, and Christopher Musco. Input sparsity time low-rank approximation via ridge leverage score sampling. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1758–1777, 2017.
- [CW17] Kenneth L. Clarkson and David P. Woodruff. Low-rank PSD approximation in input-sparsity time. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2061–2072, 2017.
- [DM05] Petros Drineas and Michael W Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.
- [DMIMW12] Petros Drineas, Malik Magdon-Ismail, Michael W. Mahoney, and David P. Woodruff. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research*, 13:3475–3506, 2012.

- [DMM08] Petros Drineas, Michael W Mahoney, and S Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, 2008.
- [DST03] Vin De Silva and Joshua B Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in Neural Information Processing Systems 16 (NIPS)*, pages 721–728, 2003.
- [FS02] Shai Fine and Katya Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2002.
- [FSS13] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k -means, PCA, and projective clustering. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1434–1453, 2013.
- [Git11] Alex Gittens. The spectral norm error of the naive Nyström extension. [arXiv:1110.5305](https://arxiv.org/abs/1110.5305), 2011.
- [GM13] Alex Gittens and Michael Mahoney. Revisiting the Nyström method for improved large-scale machine learning. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 567–575, 2013. Full version at [arXiv:1303.1849](https://arxiv.org/abs/1303.1849).
- [HFH⁺09] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.
- [HKZ14] Daniel Hsu, Sham M. Kakade, and Tong Zhang. Random design analysis of ridge regression. *Foundations of Computational Mathematics*, 14(3):569–600, 2014.
- [HTF02] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2nd edition, 2002.
- [IBM14] IBM Research Division, Skylark Team. *Libskylark: Sketching-based Distributed Matrix Computations for Machine Learning*. IBM Corporation, Armonk, NY, 2014.
- [KMT12] Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling methods for the Nyström method. *Journal of Machine Learning Research*, 13:981–1006, 2012.
- [LBKL15] Mu Li, Wei Bi, James T Kwok, and Bao-Liang Lu. Large-scale Nyström kernel matrix approximation using randomized SVD. *IEEE Transactions on Neural Networks and Learning Systems*, 26(1):152–164, 2015.
- [Lic13] M. Lichman. UCI machine learning repository, 2013.
- [LJS16] Chengtao Li, Stefanie Jegelka, and Suvrit Sra. Fast DPP sampling for Nyström with application to kernel methods. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 2016.

- [LSS13] Quoc Le, Tamás Sarlós, and Alexander Smola. Fastfood - Computing Hilbert space expansions in loglinear time. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 244–252, 2013.
- [PD16] Saurabh Paul and Petros Drineas. Feature selection for ridge regression with provable guarantees. *Neural Computation*, 28(4):716–742, 2016.
- [Pla05] John Platt. FastMap, MetricMap, and Landmark MDS are all Nyström algorithms. In *Proceedings of the 8th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2005.
- [PVG⁺11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [RCR15] Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems 28 (NIPS)*, pages 1648–1656, 2015.
- [RR07] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems 20 (NIPS)*, pages 1177–1184, 2007.
- [RR09] Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems 22 (NIPS)*, pages 1313–1320, 2009.
- [SS00] Alex J Smola and Bernhard Schölkopf. Sparse greedy matrix approximation for machine learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pages 911–918, 2000.
- [SS02] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.
- [SSM99] Bernhard Schölkopf, Alexander J. Smola, and Klaus-Robert Müller. Advances in kernel methods. chapter Kernel principal component analysis, pages 327–352. MIT Press, 1999.
- [Tro15] Joel A. Tropp. An introduction to matrix concentration inequalities. *Foundations and Trends in Machine Learning*, 8(1-2):1–230, 2015.
- [TRVR16] Stephen Tu, Rebecca Roelofs, Shivaram Venkataraman, and Benjamin Recht. Large scale kernel learning using block coordinate descent. [arXiv:1602.05310](https://arxiv.org/abs/1602.05310), 2016.
- [UKM06] Andrew V Uzilov, Joshua M Keegan, and David H Mathews. Detection of non-coding rnas on the basis of predicted secondary structure formation free energy change. *BMC bioinformatics*, 7(1):173, 2006.
- [Wan16] Weiran Wang. On column selection in approximate kernel canonical correlation analysis. [arXiv:1602.02172](https://arxiv.org/abs/1602.02172), 2016.

- [Woo14] David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.
- [WS01] Christopher Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems 14 (NIPS)*, pages 682–688, 2001.
- [WZ13] Shusen Wang and Zhihua Zhang. Improving CUR matrix decomposition and the Nyström approximation via adaptive sampling. *Journal of Machine Learning Research*, 14:2729–2769, 2013.
- [YLM⁺12] Tianbao Yang, Yu-feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi-Hua Zhou. Nyström method vs random fourier features: A theoretical and empirical comparison. In *Advances in Neural Information Processing Systems 25 (NIPS)*, pages 476–484, 2012.
- [YPW15] Yun Yang, Mert Pilanci, and Martin J Wainwright. Randomized sketches for kernels: Fast and optimal non-parametric regression. *Annals of Statistics*, 2015.
- [YZ13] Martin Wainwright Yuchen Zhang, John Duchi. Divide and conquer kernel ridge regression. *Proceedings of the 26th Annual Conference on Computational Learning Theory (COLT)*, 2013.
- [ZTK08] Kai Zhang, Ivor W. Tsang, and James T. Kwok. Improved Nyström low-rank approximation and error analysis. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pages 1232–1239, 2008.

A Ridge leverage score sampling bounds

Here we give the primary matrix concentration results used to bound the performance of ridge leverage score sampling in Theorems 3, 7, and 10.

Lemma 11. *For any $\lambda > 0$ and $\delta \in (0, 1/8)$, given ridge leverage score approximations $\tilde{l}_i^\lambda \geq l_i^\lambda$ for all i , let $p_i = \min \left\{ 1, 16\tilde{l}_i^\lambda \log(\sum \tilde{l}_i^\lambda / \delta) \right\}$. Let $\mathbf{S} \in \mathbb{R}^{n \times s}$ be selected by sampling the standard basis vectors $\mathbf{e}_1, \dots, \mathbf{e}_n$ each independently with probability p_i and rescaling selected columns by $1/\sqrt{p_i}$. With probability $1 - \delta$, $1/2 \cdot \sum_i p_i \leq s \leq 2 \sum_i p_i$ and:*

$$\frac{1}{2}\mathbf{B}^T\mathbf{B} - \frac{1}{2}\lambda\mathbf{I} \preceq \mathbf{B}^T\mathbf{S}\mathbf{S}^T\mathbf{B} \preceq \frac{3}{2}\mathbf{B}^T\mathbf{B} + \frac{1}{2}\lambda\mathbf{I}, \quad (16)$$

Proof. Let $\mathbf{B} = \mathbf{U}\Sigma\mathbf{V}^T$ be the singular value decomposition of \mathbf{B} . By Definition 1:

$$\begin{aligned} l_i^\lambda &= \mathbf{b}_i^T (\mathbf{B}^T\mathbf{B} + \lambda\mathbf{I})^{-1} \mathbf{b}_i = \mathbf{b}_i^T (\mathbf{V}\Sigma^2\mathbf{V}^T + \lambda\mathbf{V}\mathbf{V}^T)^{-1} \mathbf{b}_i \\ &= \mathbf{b}_i^T (\mathbf{V}\bar{\Sigma}^2\mathbf{V}^T)^{-1} \mathbf{b}_i \\ &= \mathbf{b}_i^T (\mathbf{V}\bar{\Sigma}^{-2}\mathbf{V}^T) \mathbf{b}_i, \end{aligned}$$

where $\bar{\Sigma}_{i,i}^2 = \sigma_i^2(\mathbf{B}) + \lambda$. For each $i \in 1, \dots, n$ define the matrix valued random variable:

$$\mathbf{X}_i = \begin{cases} \left(\frac{1}{p_i} - 1\right) \bar{\Sigma}^{-1}\mathbf{V}^T\mathbf{b}_i\mathbf{b}_i^T\mathbf{V}\bar{\Sigma}^{-1} & \text{with probability } p_i \\ -\bar{\Sigma}^{-1}\mathbf{V}^T\mathbf{b}_i\mathbf{b}_i^T\mathbf{V}\bar{\Sigma}^{-1} & \text{with probability } (1 - p_i) \end{cases}$$

Let $\mathbf{Y} = \sum_i \mathbf{X}_i$. We have $\mathbb{E} \mathbf{Y} = \mathbf{0}$. Furthermore, $\mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} = \mathbf{V} \bar{\Sigma} \mathbf{Y} \bar{\Sigma} \mathbf{V}^T + \mathbf{B}^T \mathbf{B}$. If we can show that $\|\mathbf{Y}\|_2 \leq \frac{1}{2}$, then since $\mathbf{V} \bar{\Sigma}^2 \mathbf{V}^T = \mathbf{B}^T \mathbf{B} + \lambda \mathbf{I}$ this would give the desired bound:

$$\frac{1}{2} \mathbf{B}^T \mathbf{B} - \frac{1}{2} \lambda \mathbf{I} \preceq \mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} \preceq \frac{3}{2} \mathbf{B}^T \mathbf{B} + \frac{1}{2} \lambda \mathbf{I}.$$

To prove that $\|\mathbf{Y}\|_2$ is small we use an intrinsic dimension matrix Bernstein inequality. This inequality will bound the deviation of \mathbf{Y} from its expectation as long as we can bound each $\|\mathbf{X}_i\|_2$ and we can bound the matrix variance $\mathbb{E}(\mathbf{Y}^2)$.

Theorem 12 (Theorem 7.3.1, [Tro15]). *Let $\mathbf{X}_1, \dots, \mathbf{X}_n$ be random symmetric matrices such that for all i , $\mathbb{E} \mathbf{X}_i = \mathbf{0}$ and $\|\mathbf{X}_i\|_2 \leq L$. Let $\mathbf{Y} = \sum_{i=1}^n \mathbf{X}_i$. As long we can bound the matrix variance:*

$$\mathbb{E}(\mathbf{Y}^2) \preceq \mathbf{Z},$$

then for for $t \geq \sqrt{\|\mathbf{Z}\|_2} + L/3$,

$$\mathbb{P}[\|\mathbf{Y}\| \geq t] \leq 4 \frac{\text{tr}(\mathbf{Z})}{\|\mathbf{Z}\|_2} e^{\frac{-t^2/2}{\|\mathbf{Z}\|_2 + L/3}}.$$

If $p_i = 1$ (i.e. $\tilde{c} l_i^\lambda \log(\sum l_i^\lambda / \delta) \geq 1$) then $\mathbf{X}_i = \mathbf{0}$ so $\|\mathbf{X}_i\|_2 = 0$. Otherwise, we use the fact that:

$$\frac{1}{\tilde{l}_i^\lambda} \mathbf{b}_i \mathbf{b}_i^T \preceq \frac{1}{l_i^\lambda} \mathbf{b}_i \mathbf{b}_i^T \preceq \mathbf{B}^T \mathbf{B} + \lambda \mathbf{I}. \quad (17)$$

This follows because we can write any \mathbf{x} as $\mathbf{x} = (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2} \mathbf{y}$ for some \mathbf{y} . We can then write:

$$\begin{aligned} \mathbf{x}^T \mathbf{b}_i \mathbf{b}_i^T \mathbf{x} &= \mathbf{y}^T (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2} \mathbf{b}_i \mathbf{b}_i^T (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2} \mathbf{y} \\ &\leq \|\mathbf{y}\|_2^2 \cdot \|(\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2} \mathbf{b}_i \mathbf{b}_i^T (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2}\|_2. \end{aligned}$$

Since $(\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2} \mathbf{b}_i \mathbf{b}_i^T (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2}$ is rank 1, we have:

$$\begin{aligned} \|(\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2} \mathbf{b}_i \mathbf{b}_i^T (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2}\|_2 &= \text{tr} \left((\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2} \mathbf{b}_i \mathbf{b}_i^T (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1/2} \right) \\ &= \mathbf{b}_i^T (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{-1} \mathbf{b}_i = l_i^\lambda \end{aligned} \quad (18)$$

where in the last step we use the cyclic property of the trace. Writing $\mathbf{y} = (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I})^{1/2} \mathbf{x}$ and plugging back into (18) gives:

$$\mathbf{x}^T \mathbf{b}_i \mathbf{b}_i^T \mathbf{x} \leq \|\mathbf{y}\|_2^2 \cdot l_i^\lambda = \mathbf{x}^T (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I}) \mathbf{x} \cdot l_i^\lambda.$$

Rearranging and using that $\tilde{l}_i^\lambda \geq l_i^\lambda$ gives (17). With this bound in place we get:

$$\frac{1}{\tilde{l}_i^\lambda} \cdot \bar{\Sigma}^{-1} \mathbf{V}^T \mathbf{b}_i \mathbf{b}_i^T \mathbf{V} \bar{\Sigma}^{-1} \preceq \bar{\Sigma}^{-1} \mathbf{V}^T (\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I}) \mathbf{V} \bar{\Sigma}^{-1} = \mathbf{I}.$$

So we have:

$$\mathbf{X}_i \preceq \frac{1}{p_i} \bar{\Sigma}^{-1} \mathbf{V}^T \mathbf{b}_i \mathbf{b}_i^T \mathbf{V} \bar{\Sigma}^{-1} \preceq \frac{\tilde{l}_i^\lambda}{p_i} \mathbf{I} = \frac{1}{16 \log \left(\sum l_i^\lambda / \delta \right)} \mathbf{I} \preceq \frac{1}{16 \log \left(\sum l_i^\lambda / \delta \right)} \mathbf{I}.$$

Next we bound the variance of \mathbf{Y} .

$$\begin{aligned}
\mathbb{E}(\mathbf{Y}^2) &= \sum \mathbb{E}(\mathbf{X}_i^2) \preceq \sum \left[p_i \left(\frac{1}{p_i} - 1 \right)^2 + (1 - p_i) \right] \cdot \bar{\Sigma}^{-1} \mathbf{V}^T \mathbf{b}_i \mathbf{b}_i^T \mathbf{V} \bar{\Sigma}^{-2} \mathbf{V}^T \mathbf{b}_i \mathbf{b}_i^T \mathbf{V} \bar{\Sigma}^{-1} \\
&\preceq \sum \frac{1}{p_i} \cdot l_i^\lambda \cdot \bar{\Sigma}^{-1} \mathbf{V}^T \mathbf{b}_i \mathbf{b}_i^T \mathbf{V} \bar{\Sigma}^{-1} \preceq \frac{1}{16 \log(\sum l_i^\lambda / \delta)} \bar{\Sigma}^{-1} \mathbf{V}^T \mathbf{B}^T \mathbf{B} \mathbf{V} \bar{\Sigma}^{-1} \\
&\preceq \frac{1}{16 \log(\sum l_i^\lambda / \delta)} \Sigma^2 \bar{\Sigma}^{-2} \preceq \frac{1}{16 \log(\sum l_i^\lambda / \delta)} \mathbf{D}.
\end{aligned} \tag{19}$$

where $\mathbf{D}_{1,1} = 1$ and $\mathbf{D}_{i,i} = (\Sigma^2 \bar{\Sigma}^{-2})_{i,i} = \frac{\sigma_i^2(\mathbf{B})}{\sigma_i^2(\mathbf{B}) + \lambda}$ for all $i \geq 2$. Note that $\|\mathbf{D}\|_2 = 1$.

Then applying Theorem 12 with $\mathbf{Z} = \mathbf{D} / 16 \log(\sum l_i^\lambda / \delta)$ we see that:

$$\mathbb{P} \left[\|\mathbf{Y}\|_2 \geq \frac{1}{2} \right] \leq 4 \operatorname{tr}(\mathbf{D}) e^{\frac{-1/8}{\frac{1}{16 \log(\sum l_i^\lambda / \delta)} + \frac{1}{192 \log(\sum l_i^\lambda / \delta)}}}. \tag{20}$$

Then we observe that:

$$\operatorname{tr}(\mathbf{D}) \leq 1 + \operatorname{tr}(\Sigma^2 \bar{\Sigma}^{-2}) = 1 + \operatorname{tr}(\mathbf{K}(\mathbf{K} + \lambda \mathbf{I})^{-1}) = 1 + \sum_i l_i^\lambda.$$

Plugging into (20), establishes (21):

$$\mathbb{P} \left[\|\mathbf{Y}\| \geq \frac{1}{2} \right] \leq 4 \left(1 + \sum_i l_i^\lambda \right) \cdot e^{-2 \log(\sum l_i^\lambda / \delta)} \leq \delta/2.$$

Note that here we make the extremely mild assumption that $\sum_i l_i^\lambda \geq 1$. If not, we can simply use a smaller λ that makes this condition true, and will have $s = O(1)$.

All that remains to show is that the sample size s is bounded with high probability. If $p_i = 1$, we always sample i so there is no variance in s . Let $S \subseteq [1, \dots, n]$ be the set of indices with $p_i < 1$. The expected number of points sampled from S is $\sum_{i \in S} p_i = 16 \log(\sum \tilde{l}_i^\lambda / \delta) \sum_{i \in S} \tilde{l}_i^\lambda$. Assume without loss of generality that $\sum_{i \in S} \tilde{l}_i^\lambda \geq 1$ – otherwise can just increase our leverage score estimates and increase the expected sample size by at most 1. Then, by a standard Chernoff bound, with probability at least $1 - \delta/2$,

$$\frac{1}{2} \cdot 16 \log(\sum \tilde{l}_i^\lambda / \delta) \sum_{i \in S} \tilde{l}_i^\lambda \leq s \leq 2 \cdot 16 \log(\sum \tilde{l}_i^\lambda / \delta) \sum_{i \in S} \tilde{l}_i^\lambda.$$

Union bounding over failure probabilities gives the lemma. \square

Lemma 11 yields an easy corollary about sampling *without rescaling* the columns in \mathbf{S} :

Corollary 13. *For any $\lambda > 0$ and $\delta \in (0, 1/8)$, given ridge leverage score approximations $\tilde{l}_i^\lambda \geq l_i^\lambda$ for all i , let $p_i = \min \left\{ 16 \tilde{l}_i^\lambda \log(\sum \tilde{l}_i^\lambda / \delta), 1 \right\}$. Let $\mathbf{S} \in \mathbb{R}^{n \times s}$ be selected by sampling, **but not rescaling**, the standard basis vectors $\mathbf{e}_1, \dots, \mathbf{e}_n$ each independently with probability p_i . With probability $1 - \delta$, $1/2 \cdot \sum_i p_i \leq s \leq 2 \sum_i p_i$ and there exists some scaling factor $C > 0$ such that*

$$\mathbf{B}^T \mathbf{B} \preceq C \cdot \mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda \mathbf{I}. \tag{21}$$

Proof. By Lemma 11, if we set $C' = \frac{1}{\min_i p_i}$ we have:

$$\begin{aligned}\frac{1}{2}\mathbf{B}^T\mathbf{B} - \frac{1}{2}\lambda\mathbf{I} &\preceq C' \cdot \mathbf{B}^T\mathbf{S}\mathbf{S}^T\mathbf{B} \\ \mathbf{B}^T\mathbf{B} &\preceq 2C' \cdot \mathbf{B}^T\mathbf{S}\mathbf{S}^T\mathbf{B} + \lambda\mathbf{I}\end{aligned}$$

which gives the corollary by setting $C = 2C'$. \square

B Projection-cost preserving kernel approximation

In addition to the basic spectral approximation guarantee of Theorem 3, we prove that, with high probability, the RLS-Nyström method presented in Algorithm 1 outputs an approximation $\tilde{\mathbf{K}}$ satisfying what is known as a *projection-cost preservation guarantee*. This approximation also immediately holds for the efficient implementation of sampling in Algorithm 3.

Theorem 14 (Projection-cost preserving kernel approximation). *Let $\lambda = \frac{\epsilon}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})$. For any $\epsilon \in (0, 1)$, $\delta \in (0, 1/8)$, RLS-Nyström returns an $\mathbf{S} \in \mathbb{R}^{n \times s}$ such that with probability $1 - \delta$, $1/2 \sum_i p_i \leq s \leq 2 \sum_i p_i$ and the approximation $\tilde{\mathbf{K}} = \mathbf{K}\mathbf{S}(\mathbf{S}\mathbf{K}\mathbf{S})^+\mathbf{S}\mathbf{K}$ satisfies, for any rank k orthogonal projection \mathbf{X} and a positive constant c independent of \mathbf{X} :*

$$\text{tr}(\mathbf{K} - \mathbf{X}\mathbf{K}\mathbf{X}) \leq \text{tr}(\tilde{\mathbf{K}} - \mathbf{X}\tilde{\mathbf{K}}\mathbf{X}) + c \leq (1 + \epsilon) \text{tr}(\mathbf{K} - \mathbf{X}\mathbf{K}\mathbf{X}). \quad (22)$$

When ridge leverage scores are computed exactly, $\sum_i p_i = O\left(\frac{k}{\epsilon} \log \frac{k}{\delta\epsilon}\right)$.

Intuitively, Theorem 14 ensures that the distance from $\tilde{\mathbf{K}}$ to any low dimensional subspace closely approximates the distance from \mathbf{K} to the subspace. Accordingly, $\tilde{\mathbf{K}}$ can be used in place of \mathbf{K} to approximately solve low-rank approximation problems, both constrained (e.g. k -means clustering) and unconstrained (e.g. principal component analysis). See Theorems 16 and 17.

Proof. Set $c = \text{tr}(\mathbf{K}) - \text{tr}(\tilde{\mathbf{K}})$, which is ≥ 0 since $\tilde{\mathbf{K}} \preceq \mathbf{K}$ by Theorem 3. By linearity of trace:

$$\text{tr}(\tilde{\mathbf{K}} - \mathbf{X}\tilde{\mathbf{K}}\mathbf{X}) + c = \text{tr}(\mathbf{K}) - \text{tr}(\mathbf{X}\tilde{\mathbf{K}}\mathbf{X}).$$

So to obtain (22) it suffices to show:

$$\text{tr}(\mathbf{X}\mathbf{K}\mathbf{X}) - \epsilon \text{tr}(\mathbf{K} - \mathbf{X}\mathbf{K}\mathbf{X}) \leq \text{tr}(\mathbf{X}\tilde{\mathbf{K}}\mathbf{X}) \leq \text{tr}(\mathbf{X}\mathbf{K}\mathbf{X}). \quad (23)$$

Since \mathbf{X} is a rank k orthogonal projection we can write $\mathbf{X} = \mathbf{Q}\mathbf{Q}^T$ where $\mathbf{Q} \in \mathbb{R}^{n \times k}$ has orthonormal columns. Applying the cyclic property of the trace, and the spectral bound of Theorem 3:

$$\text{tr}(\mathbf{X}\tilde{\mathbf{K}}\mathbf{X}) = \text{tr}(\mathbf{Q}^T\tilde{\mathbf{K}}\mathbf{Q}) = \sum_{i=1}^k \mathbf{q}_i^T \tilde{\mathbf{K}} \mathbf{q}_i \leq \sum_{i=1}^k \mathbf{q}_i^T \mathbf{K} \mathbf{q}_i = \text{tr}(\mathbf{Q}^T \mathbf{K} \mathbf{Q}) = \text{tr}(\mathbf{X}\mathbf{K}\mathbf{X}).$$

This gives us the upper bound of (23). For the lower bound we apply Corollary 4:

$$\text{tr}(\mathbf{X}\tilde{\mathbf{K}}\mathbf{X}) = \sum_{i=1}^k \mathbf{q}_i^T \tilde{\mathbf{K}} \mathbf{q}_i \geq \sum_{i=1}^k \mathbf{q}_i^T \mathbf{K} \mathbf{q}_i - k\epsilon\lambda = \text{tr}(\mathbf{X}\mathbf{K}\mathbf{X}) - k\epsilon\lambda. \quad (24)$$

Finally, $k\epsilon\lambda = \epsilon \sum_{i=k+1}^n \sigma_i(\mathbf{K}) \leq \epsilon \text{tr}(\mathbf{K} - \mathbf{X}\mathbf{K}\mathbf{X})$ since $\text{tr}(\mathbf{K}) = \sum_{i=1}^n \sigma_i(\mathbf{K})$ and $\text{tr}(\mathbf{X}\mathbf{K}\mathbf{X}) \leq \sum_{i=1}^k \sigma_i(\mathbf{K})$ by the Eckart-Young theorem. Plugging into (24) gives (23), completing the proof.

We conclude by showing that s is not too large. As in the proof of Theorem 3, $s \leq 2 \sum_i p_i$ with probability $1 - \delta$. When ridge leverage scores are computed exactly $\sum_i p_i \leq 16 \sum l_i^\lambda \log(\sum l_i^\lambda / \delta)$.

$$\begin{aligned}
\sum_i l_i^\lambda &= \text{tr}(\mathbf{K}(\mathbf{K} + \epsilon \left(\frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K}) \right) \mathbf{I})^{-1}) \\
&\leq \frac{1}{\epsilon} \text{tr}(\mathbf{K}(\mathbf{K} + \left(\frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K}) \right) \mathbf{I})^{-1}) \\
&= \frac{1}{\epsilon} \sum_{i=1}^n \frac{\sigma_i(\mathbf{K})}{\sigma_i(\mathbf{K}) + \frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})} \\
&= \frac{1}{\epsilon} \left(\sum_{i=1}^k \frac{\sigma_i(\mathbf{K})}{\sigma_i(\mathbf{K}) + \frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})} + \sum_{i=k+1}^n \frac{\sigma_i(\mathbf{K})}{\sigma_i(\mathbf{K}) + \frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})} \right) \\
&\leq \frac{1}{\epsilon} \left(k + \sum_{i=k+1}^n \frac{\sigma_i(\mathbf{K})}{\frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})} \right) = \frac{2k}{\epsilon}.
\end{aligned} \tag{25}$$

Accordingly, $\sum_i p_i = 32 \frac{k}{\epsilon} \log \frac{k}{\delta \epsilon}$ as desired. \square

C Applications to learning tasks

In this section we use our general approximation guarantees from Theorems 3 and 14 to prove that the kernel approximations given by RLS-Nystrom sampling are sufficient for many downstream learning tasks. In other words, $\tilde{\mathbf{K}}$ can be used in place of \mathbf{K} without sacrificing accuracy or statistical performance in the final computation.

C.1 Kernel ridge regression

We begin with a standard formulation of the ubiquitous kernel ridge regression task [SS02]. Given input data points $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \dots, y_n \in \mathbb{R}$ this problem asks us to solve:

$$\boldsymbol{\alpha} \stackrel{\text{def}}{=} \arg \min_{\mathbf{c} \in \mathbb{R}^n} \|\mathbf{K}\mathbf{c} - \mathbf{y}\|_2^2 + \lambda \mathbf{c}^T \mathbf{K} \mathbf{c}, \tag{26}$$

which can be done in closed form by computing:

$$\boldsymbol{\alpha} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y}.$$

For prediction, when we're given a new input \mathbf{x} , we evaluate its label to be:

$$y = \sum_{i=1}^n \alpha_i K(\mathbf{x}_i, \mathbf{x}). \tag{27}$$

C.1.1 Approximate kernel ridge regression

Naively, solving for α exactly requires at least $O(n^2)$ time to compute \mathbf{K} , plus the cost of a direct or iterative matrix inversion algorithm. Prediction is also costly since it requires a kernel evaluation with all n training points. These costs can be reduced significantly using Nyström approximation.

In particular, we first select landmark points and compute the kernel approximation $\tilde{\mathbf{K}} = \mathbf{KS}^T\mathbf{KS} + \mathbf{S}^T\mathbf{K}$. We can then compute an approximate set of coefficients:

$$\tilde{\alpha} \stackrel{\text{def}}{=} (\tilde{\mathbf{K}} + \lambda \mathbf{I})^{-1} \mathbf{y}. \quad (28)$$

With a direct matrix inversion, doing so only takes $O(ns^2)$ time when our sampling matrix $\mathbf{S} \in \mathbb{R}^{n \times s}$ selects s landmark points. This is a significant improvement on the $O(n^3)$ time required to invert the full kernel. Additionally, the cost of multiplying by $\tilde{\mathbf{K}} + \lambda \mathbf{I}$, which determines the cost of most iterative regression solvers, is reduced, from $O(n^2)$ to $O(ns)$.

To predict a label for a new \mathbf{x} , we first compute its kernel product with all of our landmark points. Specifically, let $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(s)}$ be the landmarks selected by \mathbf{S} 's columns. Define $\mathbf{w} \in \mathbb{R}^s$ as:

$$\mathbf{w}_i \stackrel{\text{def}}{=} K(\mathbf{x}^{(i)}, \mathbf{x}).$$

and let

$$y = \mathbf{w}^T (\mathbf{S}^T \mathbf{KS}) + \mathbf{S}^T \mathbf{K} \tilde{\alpha}. \quad (29)$$

Computationally, it makes sense to precompute $(\mathbf{S}^T \mathbf{KS}) + \mathbf{S}^T \mathbf{K} \tilde{\alpha}$. Then the cost of prediction is just s kernel evaluations to compute \mathbf{w} , plus s additional operations to multiply \mathbf{w}^T by $(\mathbf{S}^T \mathbf{KS}) + \mathbf{S}^T \mathbf{K} \tilde{\alpha}$.

This approach is the standard way of applying Nyström approximation to the ridge regression problem and there are a number of ways to evaluate its performance. Beyond directly bounding minimization error for (26) (see e.g. [CLL⁺15, YPW15, YZ13]), one particularly natural approach is to consider how the statistical risk of the estimator output by our approximate ridge regression routine compares to that of the exactly computed estimator.

C.1.2 Relative error bound on statistical risk

To evaluate statistical risk we consider a *fixed design* setting which has been especially -popular [Bac13, AM15, LJS16, PD16]. Note that more complex statistical models can be analyzed as well [HKZ14, RCR15]. In this setting, we assume that our observed labels $\mathbf{y} = [y_1, \dots, y_n]$ represent underlying true labels $\mathbf{z} = [z_1, \dots, z_n]$ perturbed with noise. For simplicity, we assume uniform Gaussian noise with variance σ^2 , but more general noise models can be handled with essentially the same proof [Bac13]. In particular, our modeling assumption is that:

$$y_i = z_i + \eta_i$$

where $\eta_i \sim N(0, \sigma^2)$.

Following, [Bac13] and [AM15], we want to bound the expected in sample risk of our estimator for \mathbf{z} , which is computed using the noisy measurements $\mathbf{y} = \mathbf{z} + \boldsymbol{\eta}$. For exact kernel ridge regression,

we can check from (27) that this estimator is equal to $\mathbf{K}\boldsymbol{\alpha}$. The risk \mathcal{R} is:

$$\begin{aligned}\mathcal{R} &\stackrel{\text{def}}{=} \mathbb{E}_{\boldsymbol{\eta}} \|\mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1}(\mathbf{z} + \boldsymbol{\eta}) - \mathbf{z}\|_2^2 \\ &= \|(\mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1} - \mathbf{I})\mathbf{z}\|_2^2 + \mathbb{E}_{\boldsymbol{\eta}} \|\mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1}\boldsymbol{\eta}\|_2^2 \\ &= \lambda^2 \mathbf{z}^T (\mathbf{K} + \lambda\mathbf{I})^{-2} \mathbf{z} + \sigma^2 \text{tr}(\mathbf{K}^2 (\mathbf{K} + \lambda\mathbf{I})^{-2}).\end{aligned}$$

The two terms that compose \mathcal{R} are referred to as the bias and variance terms of the risk:

$$\begin{aligned}\text{bias}(\mathbf{K})^2 &\stackrel{\text{def}}{=} \lambda^2 \mathbf{z}^T (\mathbf{K} + \lambda\mathbf{I})^{-2} \mathbf{z} \\ \text{variance}(\mathbf{K}) &\stackrel{\text{def}}{=} \sigma^2 \text{tr}(\mathbf{K}^2 (\mathbf{K} + \lambda\mathbf{I})^{-2}).\end{aligned}$$

For approximate kernel ridge regression, it follows from (29) that our predictor for \mathbf{z} is $\tilde{\mathbf{K}}\tilde{\boldsymbol{\alpha}}$. Accordingly, the risk of the approximate estimator, $\tilde{\mathcal{R}}$ is equal to:

$$\tilde{\mathcal{R}} = \text{bias}(\tilde{\mathbf{K}})^2 + \text{variance}(\tilde{\mathbf{K}})$$

We're ready to prove our main theorem on kernel ridge regression.

Theorem 15 (Kernel Ridge Regression Risk Bound). *Suppose $\tilde{\mathbf{K}}$ is computed using RLS-Nystrom with approximation parameter $\epsilon\lambda$ and failure probability $\delta \in (0, 1/8)$. Let $\tilde{\boldsymbol{\alpha}} = (\tilde{\mathbf{K}} + \lambda\mathbf{I})^{-1}\mathbf{y}$ and let $\tilde{\mathbf{K}}\tilde{\boldsymbol{\alpha}}$ be our estimator for \mathbf{z} computed with the approximate kernel. With probability $1 - \delta$:*

$$\tilde{\mathcal{R}} \leq (1 + 3\epsilon)\mathcal{R}.$$

By Theorem 7, Algorithm 2 can compute $\tilde{\mathbf{K}}$ with just $O(ns)$ kernel evaluations and $O(ns^2)$ computation time, with $s = O\left(\frac{d_{\text{eff}}^\lambda}{\epsilon} \log \frac{d_{\text{eff}}^\lambda}{\delta\epsilon}\right)$.

In other words, replacing \mathbf{K} with the approximation $\tilde{\mathbf{K}}$ is provably sufficient for obtaining a $1 + \Theta(\epsilon)$ quality solution to the downstream task of ridge regression.

Proof. The proof follows that of Theorem 1 in [AM15]. First we show that:

$$\text{bias}(\tilde{\mathbf{K}}) \leq (1 + \epsilon)\text{bias}(\mathbf{K}). \quad (30)$$

At first glance this might appear trivial as Theorem 3 easily implies that

$$(\tilde{\mathbf{K}} + \lambda\mathbf{I})^{-1} \preceq (1 + \epsilon)(\mathbf{K} + \lambda\mathbf{I})^{-1}$$

However, this statement *does not imply* that

$$(\tilde{\mathbf{K}} + \lambda\mathbf{I})^{-2} \preceq (1 + \epsilon)^2 (\mathbf{K} + \lambda\mathbf{I})^{-2}$$

since $(\tilde{\mathbf{K}} + \lambda\mathbf{I})^{-1}$ and $(\mathbf{K} + \lambda\mathbf{I})^{-1}$ do not necessarily commute. Instead we proceed:

$$\begin{aligned}\frac{1}{\lambda}\text{bias}(\tilde{\mathbf{K}}) &= \|(\tilde{\mathbf{K}} + \lambda\mathbf{I})^{-1}\mathbf{z}\|_2 \\ &\leq \|(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{z}\|_2 + \|(\tilde{\mathbf{K}} + \lambda\mathbf{I})^{-1}\mathbf{z} - (\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{z}\|_2 && \text{(triangle inequality)} \\ &= \|(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{z}\|_2 + \|(\tilde{\mathbf{K}} + \lambda\mathbf{I})^{-1}[(\mathbf{K} + \lambda\mathbf{I}) - (\tilde{\mathbf{K}} + \lambda\mathbf{I})](\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{z}\|_2 \\ &= \|(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{z}\|_2 + \|(\tilde{\mathbf{K}} + \lambda\mathbf{I})^{-1}(\mathbf{K} - \tilde{\mathbf{K}})(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{z}\|_2 \\ &\leq \|(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{z}\|_2 + \|(\tilde{\mathbf{K}} + \lambda\mathbf{I})^{-1}(\mathbf{K} - \tilde{\mathbf{K}})\|_2 \|(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{z}\|_2 && \text{(submultiplicativity)} \\ &= \frac{1}{\lambda}\text{bias}(\mathbf{K}) \left(1 + \|(\tilde{\mathbf{K}} + \lambda\mathbf{I})^{-1}(\mathbf{K} - \tilde{\mathbf{K}})\|_2\right).\end{aligned} \quad (31)$$

So we just need to bound $\|(\tilde{\mathbf{K}} + \lambda \mathbf{I})^{-1}(\mathbf{K} - \tilde{\mathbf{K}})\|_2 \leq \epsilon$. First note that, by Theorem 3, Corollary 4,

$$\mathbf{K} - \tilde{\mathbf{K}} \preceq \epsilon \lambda \mathbf{I}$$

and since $(\mathbf{K} - \tilde{\mathbf{K}})$ and \mathbf{I} commute, it follows that

$$(\mathbf{K} - \tilde{\mathbf{K}})^2 \preceq \epsilon^2 \lambda^2 \mathbf{I}. \quad (32)$$

Accordingly,

$$\begin{aligned} \|(\tilde{\mathbf{K}} + \lambda \mathbf{I})^{-1}(\mathbf{K} - \tilde{\mathbf{K}})\|_2^2 &= \|(\tilde{\mathbf{K}} + \lambda \mathbf{I})^{-1}(\mathbf{K} - \tilde{\mathbf{K}})^2(\tilde{\mathbf{K}} + \lambda \mathbf{I})^{-1}\|_2 \\ &\leq \epsilon^2 \lambda^2 \|(\tilde{\mathbf{K}} + \lambda \mathbf{I})^{-2}\|_2 \\ &\leq \epsilon^2 \lambda^2 \frac{1}{\lambda^2} = \epsilon^2. \end{aligned}$$

So $\|(\tilde{\mathbf{K}} + \lambda \mathbf{I})^{-1}(\mathbf{K} - \tilde{\mathbf{K}})\|_2 \leq \epsilon$ as desired and plugging into (31) we have shown (30), that $\text{bias}(\tilde{\mathbf{K}}) \leq (1 + \epsilon) \text{bias}(\mathbf{K})$. We next show that:

$$\text{variance}(\tilde{\mathbf{K}}) \leq \text{variance}(\mathbf{K}), \quad (33)$$

where $\text{variance}(\mathbf{K}) = \sigma^2 \text{tr}(\mathbf{K}^2(\mathbf{K} + \lambda \mathbf{I})^{-2}) = \sigma^2 \sum_{i=1}^n \left(\frac{\sigma_i(\mathbf{K})}{\sigma_i(\mathbf{K}) + \lambda} \right)^2$. Since $\tilde{\mathbf{K}} \preceq \mathbf{K}$ by Theorem 3, $\sigma_i(\tilde{\mathbf{K}}) \leq \sigma_i(\mathbf{K})$ for all i . It follows that, for every i ,

$$\frac{\sigma_i(\tilde{\mathbf{K}})}{\sigma_i(\tilde{\mathbf{K}}) + \lambda} \leq \frac{\sigma_i(\mathbf{K})}{\sigma_i(\mathbf{K}) + \lambda}.$$

This in turn implies that

$$\sum_{i=1}^n \left(\frac{\sigma_i(\tilde{\mathbf{K}})}{\sigma_i(\tilde{\mathbf{K}}) + \lambda} \right)^2 \leq \sum_{i=1}^n \left(\frac{\sigma_i(\mathbf{K})}{\sigma_i(\mathbf{K}) + \lambda} \right)^2,$$

which gives (33). Combining (33) and (30) we conclude that, for $\epsilon < 1$,

$$\mathcal{R}(\hat{f}_{\tilde{\mathbf{K}}}) \leq (1 + \epsilon)^2 \mathcal{R}(\hat{f}_{\mathbf{K}}) \leq (1 + 3\epsilon) \mathcal{R}(\hat{f}_{\mathbf{K}}).$$

□

C.2 Kernel k -means

Kernel k -means clustering asks us to partition $\mathbf{x}_1, \dots, \mathbf{x}_n$, into k cluster sets, $\{C_1, \dots, C_k\}$. Let $\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\mathbf{x}_j \in C_i} \phi(\mathbf{x}_j)$ be the centroid of the vectors in C_i after mapping to kernel space. The goal is to choose $\{C_1, \dots, C_k\}$ which minimize the objective:

$$\sum_{i=1}^k \sum_{\mathbf{x}_j \in C_i} \|\phi(\mathbf{x}_j) - \boldsymbol{\mu}_i\|_{\mathcal{F}}^2 \quad (34)$$

It is well known that this optimization problem can be rewritten as a *constrained* low-rank approximation problem (see e.g. [BMD09] or [CEM⁺15]). In particular, for any clustering $C =$

$\{C_1, \dots, C_k\}$ we can define a rank k orthonormal matrix $\mathbf{C} \in \mathbb{R}^{n \times k}$ called the cluster indicator matrix for C . $\mathbf{C}_{i,j} = 1/\sqrt{|C_j|}$ if \mathbf{x}_i is assigned to C_j and $\mathbf{C}_{i,j} = 0$ otherwise. $\mathbf{C}^T \mathbf{C} = \mathbf{I}$, so $\mathbf{C} \mathbf{C}^T$ is a rank k projection matrix. Furthermore, it's not hard to check that:

$$\sum_{i=1}^k \sum_{\mathbf{x}_j \in C_i} \|\phi(\mathbf{x}_j) - \boldsymbol{\mu}_i\|_{\mathcal{F}}^2 = \text{tr}(\mathbf{K} - \mathbf{C} \mathbf{C}^T \mathbf{K} \mathbf{C} \mathbf{C}^T). \quad (35)$$

Informally, if we work with the kernelized data matrix Φ , (35) is equivalent to

$$\|\Phi - \mathbf{C} \mathbf{C}^T \Phi\|_F^2.$$

Regardless, it's clear that solving kernel k -means is equivalent to solving:

$$\min_{\mathbf{C} \in \mathcal{S}} \text{tr}(\mathbf{K} - \mathbf{C} \mathbf{C}^T \mathbf{K} \mathbf{C} \mathbf{C}^T) \quad (36)$$

where \mathcal{S} is the set of all rank k cluster indicator matrices. From this formulation, we easily obtain:

Theorem 16 (Kernel k -means Approximation Bound). *Let $\tilde{\mathbf{K}}$ be computed by RLS-Nyström with $\lambda = \frac{\epsilon}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})$ and $\delta \in (0, 1/8)$. Let $\tilde{\mathbf{C}}^*$ be the optimal cluster indicator matrix for $\tilde{\mathbf{K}}$ and let $\tilde{\mathbf{C}}$ be an approximately optimal cluster indicator matrix satisfying:*

$$\text{tr}(\tilde{\mathbf{K}} - \tilde{\mathbf{C}} \tilde{\mathbf{C}}^T \tilde{\mathbf{K}} \tilde{\mathbf{C}} \tilde{\mathbf{C}}^T) \leq (1 + \gamma) \text{tr}(\tilde{\mathbf{K}} - \tilde{\mathbf{C}}^* \tilde{\mathbf{C}}^{*T} \tilde{\mathbf{K}} \tilde{\mathbf{C}}^* \tilde{\mathbf{C}}^{*T}).$$

Then, if \mathbf{C}^ is the optimal cluster indicator matrix for \mathbf{K} :*

$$\text{tr}(\mathbf{K} - \tilde{\mathbf{C}} \tilde{\mathbf{C}}^T \mathbf{K} \tilde{\mathbf{C}} \tilde{\mathbf{C}}^T) \leq (1 + \gamma)(1 + \epsilon) \text{tr}(\mathbf{K} - \mathbf{C}^* \mathbf{C}^{*T} \mathbf{K} \mathbf{C}^* \mathbf{C}^{*T})$$

By Theorem 10, Algorithm 3 can compute $\tilde{\mathbf{K}}$ with $O(ns)$ kernel evaluations and $O(ns^2)$ computation time, with $s = O\left(\frac{k}{\epsilon} \log \frac{k}{\delta \epsilon}\right)$.

In other words, if we find an optimal set of clusters for our approximate kernel matrix, those clusters will provide a $(1 + \epsilon)$ approximation to the original kernel k -means problem. Furthermore, if we only solve the kernel k -means problem approximately on $\tilde{\mathbf{K}}$, i.e. with some approximation factor $(1 + \gamma)$, we will do nearly as well on the original problem. This flexibility allows for the use of k -means approximation algorithms (since the problem is NP-hard to solve exactly).

Proof. The proof is almost immediate from our bounds on RLS-Nyström:

$$\begin{aligned} \text{tr}(\mathbf{K} - \tilde{\mathbf{C}} \tilde{\mathbf{C}}^T \mathbf{K} \tilde{\mathbf{C}} \tilde{\mathbf{C}}^T) &\leq \text{tr}(\tilde{\mathbf{K}} - \tilde{\mathbf{C}} \tilde{\mathbf{C}}^T \tilde{\mathbf{K}} \tilde{\mathbf{C}} \tilde{\mathbf{C}}^T) + c && \text{(Theorem 14)} \\ &\leq (1 + \gamma) \text{tr}(\tilde{\mathbf{K}} - \tilde{\mathbf{C}}^* \tilde{\mathbf{C}}^{*T} \tilde{\mathbf{K}} \tilde{\mathbf{C}}^* \tilde{\mathbf{C}}^{*T}) + (1 + \gamma)c && \text{(by assumption)} \\ &\leq (1 + \gamma) \text{tr}(\tilde{\mathbf{K}} - \mathbf{C}^* \mathbf{C}^{*T} \tilde{\mathbf{K}} \mathbf{C}^* \mathbf{C}^{*T}) + (1 + \gamma)c && \text{(optimality of } \tilde{\mathbf{C}}^*) \\ &\leq (1 + \gamma) \text{tr}(\tilde{\mathbf{K}} - \mathbf{C}^* \mathbf{C}^{*T} \tilde{\mathbf{K}} \mathbf{C}^* \mathbf{C}^{*T}) + c && \text{(since } c \geq 0) \\ &\leq (1 + \gamma)(1 + \epsilon) \text{tr}(\mathbf{K} - \mathbf{C}^* \mathbf{C}^{*T} \mathbf{K} \mathbf{C}^* \mathbf{C}^{*T}). && \text{(Theorem 14)} \end{aligned}$$

□

C.3 Kernel principal component analysis

We consider the standard formulation of kernel principal component analysis (PCA) presented in [SSM99]. The goal is to find principal components *in the kernel space* \mathcal{F} that capture as much variance in the kernelized data as possible. In particular, if we work informally with the kernelized data matrix Φ , we want to find a matrix \mathbf{Z}_k containing k orthonormal columns such that:

$$\Phi\Phi^T - (\Phi\mathbf{Z}_k\mathbf{Z}_k^T)(\Phi\mathbf{Z}_k\mathbf{Z}_k^T)^T$$

is as small as possible. In other words, if we project Φ 's rows to the k dimensional subspace spanned by \mathbf{V}_k 's columns and then recompute our kernel, we want the approximate kernel to be close to the original.

We focus in particular on minimizing PCA error according to the metric:

$$\text{tr}(\Phi\Phi^T - (\Phi\mathbf{Z}_k\mathbf{Z}_k^T)(\Phi\mathbf{Z}_k\mathbf{Z}_k^T)^T) = \|\Phi - \Phi\mathbf{Z}_k\mathbf{Z}_k^T\|_F^2, \quad (37)$$

which is standard in the literature [Woo14, ANW14]. As with f in kernel ridge regression, to solve this problem we cannot write down \mathbf{Z}_k explicitly for most kernel functions. However, the optimal \mathbf{Z}_k always lies in the column span of Φ^T , so we can implicitly represent it by constructing a matrix $\mathbf{X} \in \mathbb{R}^{n \times k}$ such that $\Phi^T \mathbf{X} = \mathbf{Z}_k$. It is then easy to compute the projection of any new data vector onto the span of \mathbf{Z}_k (the typical objective of principal component analysis) since we can multiply by $\Phi^T \mathbf{X}$ using the kernel function.

By the Eckart-Young theorem the optimal \mathbf{Z}_k contains the top k row principal components of Φ . Accordingly, if we write the singular value decomposition $\Phi = \mathbf{U}\Sigma\mathbf{V}^T$ we want to set $\mathbf{X} = \mathbf{U}_k\Sigma_k^{-1}$, which can be computed from the SVD of $\mathbf{K} = \mathbf{U}\Sigma^2\mathbf{U}^T$. \mathbf{Z}_k will equal \mathbf{V}_k and (37) reduces to:

$$\begin{aligned} \text{tr}(\mathbf{K} - \Phi\mathbf{V}_k\mathbf{V}_k^T\Phi) &= \text{tr}(\mathbf{K} - \mathbf{V}_k\mathbf{V}_k^T\mathbf{K}) && \text{(cyclic property)} \\ &= \sum_{i=k+1}^n \sigma_i(\mathbf{K}). \end{aligned} \quad (38)$$

Theorem 17 (Kernel PCA Approximation Bound). *Let $\tilde{\mathbf{K}}$ be computed by RLS-Nyström with $\lambda = \frac{\epsilon}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})$ and $\delta \in (0, 1/8)$. From $\tilde{\mathbf{K}}$ we can compute a matrix $\mathbf{X} \in \mathbb{R}^{s \times k}$ such that if we set $\mathbf{Z} = \Phi^T \mathbf{S}\mathbf{X}$, with probability $1 - \delta$:*

$$\|\Phi - \Phi\mathbf{Z}\mathbf{Z}^T\|_F^2 \leq (1 + 2\epsilon)\|\Phi - \Phi\mathbf{V}_k\mathbf{V}_k^T\|_F^2 = (1 + 2\epsilon) \sum_{i=k+1}^n \sigma_i(\mathbf{K}).$$

By Theorem 10, Algorithm 3 can compute $\tilde{\mathbf{K}}$ with $O(ns)$ kernel evaluations and $O(ns^2)$ computation time, with $s = O\left(\frac{k}{\epsilon} \log \frac{k}{\delta\epsilon}\right)$.

Note that \mathbf{S} is the sampling matrix used to construct $\tilde{\mathbf{K}}$. $\mathbf{Z} = \Phi^T \mathbf{S}\mathbf{X}$ can be applied to vectors (in order to project onto the approximate low-rank subspace) using only s kernel evaluations.

Proof. Re-parameterizing $\mathbf{Z}_k = \Phi^T \mathbf{Y}$, we see that minimizing (37) is equivalent to minimizing

$$\text{tr}(\mathbf{K} - \mathbf{K}\mathbf{Y}\mathbf{Y}^T\mathbf{K})$$

over $\mathbf{Y} \in \mathbb{R}^{n \times k}$ such that $(\Phi^T \mathbf{Y})^T \Phi^T \mathbf{Y} = \mathbf{Y}^T \mathbf{K} \mathbf{Y} = \mathbf{I}$. Then we re-parameterize again by writing $\mathbf{Y} = \mathbf{K}^{-1/2} \mathbf{W}$ where \mathbf{W} is an $n \times k$ matrix with orthonormal columns. Using linearity and cyclic property of the trace, we can write:

$$\text{tr}(\mathbf{K} - \mathbf{K} \mathbf{Y} \mathbf{Y}^T \mathbf{K}) = \text{tr}(\mathbf{K}) - \text{tr}(\mathbf{Y}^T \mathbf{K} \mathbf{Y}) = \text{tr}(\mathbf{K}) - \text{tr}(\mathbf{W}^T \mathbf{K} \mathbf{W}) = \text{tr}(\mathbf{K}) - \text{tr}(\mathbf{W} \mathbf{W}^T \mathbf{K} \mathbf{W} \mathbf{W}^T).$$

So, we have reduced our problem to a low-rank approximation problem that looks exactly like the k -means problem from Section C.2, except without constraints.

Accordingly, following the same argument as Theorem 16, if we find $\tilde{\mathbf{W}}$ minimizing:

$$\text{tr}(\tilde{\mathbf{K}}) - \text{tr}(\tilde{\mathbf{W}} \tilde{\mathbf{W}}^T \tilde{\mathbf{K}} \tilde{\mathbf{W}} \tilde{\mathbf{W}}^T),$$

then:

$$\text{tr}(\mathbf{K}) - \text{tr}(\tilde{\mathbf{W}} \tilde{\mathbf{W}}^T \mathbf{K} \tilde{\mathbf{W}} \tilde{\mathbf{W}}^T) \leq (1 + \epsilon) \left[\min_{\mathbf{W}} \text{tr}(\mathbf{K}) - \text{tr}(\mathbf{W} \mathbf{W}^T \mathbf{K} \mathbf{W} \mathbf{W}^T) \right] = (1 + \epsilon) \sum_{i=k+1}^n \sigma_i(\mathbf{K}).$$

$\tilde{\mathbf{W}}$ can be taken to equal the top k eigenvectors of $\tilde{\mathbf{K}}$, which can be found in $O(n \cdot s^2)$ time.

However, we are not quite done. Thanks to our re-parameterization this bound guarantees that $\Phi^T \mathbf{K}^{-1/2} \tilde{\mathbf{W}}$ is a good set of approximate kernel principal components for Φ . Unfortunately, $\Phi^T \mathbf{K}^{-1/2} \tilde{\mathbf{W}}$ cannot be represented efficiently (it requires computing $\mathbf{K}^{-1/2}$) and projecting new vectors to $\Phi^T \mathbf{K}^{-1/2} \tilde{\mathbf{W}}$ would require n kernel evaluations to multiply by Φ^T .

Instead, recalling the definition of $\mathbf{P}_S = \Phi^T \mathbf{S} (\mathbf{S}^T \mathbf{K}^T \mathbf{S})^+ \mathbf{S}^T \Phi$ from Section 2.1, we suggest using the approximate principal components:

$$\mathbf{P}_S \Phi^T \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{W}}.$$

Clearly $\mathbf{P}_S \Phi^T \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{W}}$ is orthonormal because:

$$\begin{aligned} (\mathbf{P}_S \Phi^T \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{W}})^T \mathbf{P}_S \Phi^T \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{W}} &= \tilde{\mathbf{W}}^T \tilde{\mathbf{K}}^{-1/2} \Phi^T \mathbf{P}_S \Phi \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{W}} \\ &= \tilde{\mathbf{W}}^T \mathbf{I} \tilde{\mathbf{W}} = \mathbf{I}. \end{aligned}$$

We will argue that it offers nearly as good of a solution as $\Phi^T \mathbf{K}^{-1/2} \tilde{\mathbf{W}}$. Specifically, substituting into (37) gives a value of:

$$\begin{aligned} \text{tr}(\mathbf{K} - \Phi \mathbf{P}_S \Phi^T \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{W}} \tilde{\mathbf{W}}^T \tilde{\mathbf{K}}^{-1/2} \Phi \mathbf{P}_S \Phi^T) &= \text{tr}(\mathbf{K}) - \text{tr}(\tilde{\mathbf{W}} \tilde{\mathbf{W}}^T \tilde{\mathbf{K}}^{-1/2} \Phi \mathbf{P}_S \Phi^T \Phi \mathbf{P}_S \Phi^T \tilde{\mathbf{K}}^{-1/2}) \\ &= \text{tr}(\mathbf{K}) - \text{tr}(\tilde{\mathbf{W}} \tilde{\mathbf{W}}^T \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{K}}^2 \tilde{\mathbf{K}}^{-1/2}) \\ &= \text{tr}(\mathbf{K}) - \text{tr}(\tilde{\mathbf{W}} \tilde{\mathbf{W}}^T \tilde{\mathbf{K}}). \end{aligned}$$

Compare this to the value obtained from $\Phi^T \mathbf{K}^{-1/2} \tilde{\mathbf{W}}$:

$$\begin{aligned} &\left[\text{tr}(\mathbf{K}) - \text{tr}(\tilde{\mathbf{W}} \tilde{\mathbf{W}}^T \mathbf{K} \tilde{\mathbf{W}} \tilde{\mathbf{W}}^T) \right] - \left[\text{tr}(\mathbf{K}) - \text{tr}(\tilde{\mathbf{W}} \tilde{\mathbf{W}}^T \tilde{\mathbf{K}} \tilde{\mathbf{W}} \tilde{\mathbf{W}}^T) \right] \\ &= \text{tr} \left(\tilde{\mathbf{W}} \tilde{\mathbf{W}}^T (\mathbf{K} - \tilde{\mathbf{K}}) \right) = \text{tr} \left(\tilde{\mathbf{W}}^T (\mathbf{K} - \tilde{\mathbf{K}}) \tilde{\mathbf{W}} \right) = \sum_{i=1}^k \tilde{\mathbf{w}}_i^T (\mathbf{K} - \tilde{\mathbf{K}}) \tilde{\mathbf{w}}_i \leq k \frac{\epsilon}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K}). \quad (39) \end{aligned}$$

The last step follows from Theorem 3 which guarantees that $(\mathbf{K} - \tilde{\mathbf{K}}) \preceq \epsilon \lambda \mathbf{I}$. Recall that we set $\lambda = \frac{\epsilon}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})$ and each column $\tilde{\mathbf{w}}_i$ of $\tilde{\mathbf{W}}$ has unit norm.

We conclude that the cost obtained by $\mathbf{P}_S \Phi^T \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{W}}$ is bounded by:

$$\begin{aligned} \text{tr}(\mathbf{K} - \Phi \mathbf{P}_S \Phi^T \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{W}} \tilde{\mathbf{W}}^T \tilde{\mathbf{K}}^{-1/2} \Phi \mathbf{P}_S \Phi^T) &\leq \text{tr}(\mathbf{K}) - \text{tr}(\tilde{\mathbf{W}} \tilde{\mathbf{W}}^T \mathbf{K} \tilde{\mathbf{W}} \tilde{\mathbf{W}}^T) + \epsilon \sum_{i=k+1}^n \sigma_i(\mathbf{K}) \\ &\leq (1 + 2\epsilon) \sum_{i=k+1}^n \sigma_i(\mathbf{K}). \end{aligned}$$

This gives the result. Notice that $\mathbf{P}_S \Phi^T \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{W}} = \Phi^T \mathbf{S}(\mathbf{S}^T \mathbf{K}^T \mathbf{S})^+ \mathbf{S}^T \Phi \Phi^T \tilde{\mathbf{K}}^{-1/2} \tilde{\mathbf{W}}$ so, if we set:

$$\mathbf{X} = (\mathbf{S}^T \mathbf{K}^T \mathbf{S})^+ \mathbf{S}^T \tilde{\mathbf{K}}^{1/2} \tilde{\mathbf{W}},$$

our solution can be represented as $\mathbf{Z} = \Phi^T \mathbf{S} \mathbf{X}$ as desired. \square

C.4 Kernel canonical correlation analysis

We briefly discuss a final application to canonical correlation analysis (CCA) that follows from applying our spectral approximation guarantee of Theorem 3 to recent work in [Wan16].

Consider n pairs of input points $(\mathbf{x}_1, \mathbf{y}_1), \dots, (\mathbf{x}_n, \mathbf{y}_n) \in (\mathcal{X}, \mathcal{Y})$ along with two positive semidefinite kernels, $K_x : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ and $K_y : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$. Let \mathcal{F}_x and \mathcal{F}_y and $\phi_x : \mathcal{X} \rightarrow \mathcal{F}_x$ and $\phi_y : \mathcal{Y} \rightarrow \mathcal{F}_y$ be the Hilbert spaces and feature maps associated with these kernels. Let Φ_x and Φ_y denote the kernelized \mathcal{X} and \mathcal{Y} inputs respectively and \mathbf{K}_x and \mathbf{K}_y denote the associated kernel matrices.

We consider standard regularized kernel CCA, following the presentation in [Wan16]. The goal is to compute coefficient vectors α^x and α^y such that $\mathbf{f}_x^* = \sum_{i=1}^n \alpha_i^x \phi_x(\mathbf{x}_i)$ and $\mathbf{f}_y^* = \sum_{i=1}^n \alpha_i^y \phi_y(\mathbf{y}_i)$ satisfy:

$$\begin{aligned} (\mathbf{f}_x^*, \mathbf{f}_y^*) &= \arg \max_{\mathbf{f}_x \in \mathcal{F}_x, \mathbf{f}_y \in \mathcal{F}_y} \mathbf{f}_x^T \Phi_x^T \Phi_y \mathbf{f}_y^* \\ &\text{subject to} \\ \mathbf{f}_x^T \Phi_x^T \Phi_x \mathbf{f}_x + \lambda_x \|\mathbf{f}_x\|_{\mathcal{F}_x}^2 &= 1 \\ \mathbf{f}_y^T \Phi_y^T \Phi_y \mathbf{f}_y + \lambda_y \|\mathbf{f}_y\|_{\mathcal{F}_y}^2 &= 1 \end{aligned}$$

In [Wan16], the kernelized points are centered to their means. For simplicity we ignore centering, but note that [Wan16] shows how bounds for the uncentered problem carry over to the centered one.

It can be shown that $\alpha^x = (\mathbf{K}_x + \lambda_x \mathbf{I})^{-1} \beta^x$ and $\alpha^y = (\mathbf{K}_y + \lambda_y \mathbf{I})^{-1} \beta^y$ where β^x and β^y are the top left and right singular vectors respectively of

$$\mathbf{T} = (\mathbf{K}_x + \lambda_x \mathbf{I})^{-1} \mathbf{K}_x \mathbf{K}_y (\mathbf{K}_y + \lambda_y \mathbf{I})^{-1}.$$

The optimum value of the above program will be equal to $\sigma_1(\mathbf{T})$.

[Wan16] shows that if $\tilde{\mathbf{K}}_x$ and $\tilde{\mathbf{K}}_y$ satisfy:

$$\begin{aligned} \tilde{\mathbf{K}}_x &\preceq \mathbf{K}_x \preceq \tilde{\mathbf{K}}_x + \epsilon \lambda_x \mathbf{I} \\ \tilde{\mathbf{K}}_y &\preceq \mathbf{K}_y \preceq \tilde{\mathbf{K}}_y + \epsilon \lambda_y \mathbf{I} \end{aligned}$$

then if $\tilde{\alpha}^x$ and $\tilde{\alpha}^y$ are computed using these approximations, the achieved objective function value will be within ϵ of optimal (see their Lemma 1 and Theorem 1). So we have:

Theorem 18 (Kernel CCA Approximation Bound). *Suppose $\tilde{\mathbf{K}}_x$ and $\tilde{\mathbf{K}}_y$ are computed by RLS-Nyström with approximation parameters $\epsilon\lambda_x$ and $\epsilon\lambda_y$ and failure probability $\delta \in (0, 1/8)$. If we solve for $\tilde{\alpha}^x$ and $\tilde{\alpha}^y$, the approximate canonical correlation will be within an additive ϵ of the true canonical correlation $\sigma_1(\mathbf{T})$.*

By Theorem 7, Algorithm 2 can compute $\tilde{\mathbf{K}}_x$ and $\tilde{\mathbf{K}}_y$ with $O(ns_x + ns_y)$ kernel evaluations and $O(ns_x^2 + ns_y^2)$ computation time, with $s_x = O\left(\frac{d_{\text{eff}}^{\lambda_x}}{\epsilon} \log \frac{d_{\text{eff}}^{\lambda_x}}{\delta\epsilon}\right)$ and $s_y = O\left(\frac{d_{\text{eff}}^{\lambda_y}}{\epsilon} \log \frac{d_{\text{eff}}^{\lambda_y}}{\delta\epsilon}\right)$.

D Additional proofs

D.1 Effective dimension bound

Lemma 19. *For any $\mathbf{W} \in \mathbb{R}^{n \times p}$ with $\mathbf{W}\mathbf{W}^T \preceq \mathbf{I}$,*

$$\sum_{i=1}^n l_i^\lambda(\mathbf{W}^T \mathbf{K} \mathbf{W}) \leq \sum_{i=1}^n l_i^\lambda(\mathbf{K}),$$

or equivalently, by Fact 2,

$$d_{\text{eff}}^\lambda(\mathbf{W}^T \mathbf{K} \mathbf{W}) \leq d_{\text{eff}}^\lambda(\mathbf{K}).$$

Proof. By Definition 1, $l_i^\lambda = (\mathbf{K}(\mathbf{K} + \lambda \mathbf{I})^{-1})_{i,i}$ so

$$\sum_{i=1}^n l_i^\lambda(\mathbf{K}) = \text{tr}(\mathbf{K}(\mathbf{K} + \lambda \mathbf{I})^{-1}) = \sum_{i=1}^n \frac{\sigma_i(\mathbf{K})}{\sigma_i(\mathbf{K}) + \lambda}.$$

Take any matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$ such that $\mathbf{B}\mathbf{B}^T = \mathbf{K}$. Note that for any matrix \mathbf{Y} , $\sigma_i(\mathbf{Y}\mathbf{Y}^T) = \sigma_i(\mathbf{Y}^T\mathbf{Y})$ for any non-zero singular values. Accordingly,

$$\sigma_i(\mathbf{W}^T \mathbf{K} \mathbf{W}) = \sigma_i(\mathbf{W}^T \mathbf{B} \mathbf{B}^T \mathbf{W}) = \sigma_i(\mathbf{B}^T \mathbf{W} \mathbf{W}^T \mathbf{B}) \leq \sigma_i(\mathbf{B}^T \mathbf{B}) = \sigma_i(\mathbf{B} \mathbf{B}^T) = \sigma_i(\mathbf{K})$$

The \leq step follows from $\mathbf{W}\mathbf{W}^T \preceq \mathbf{I}$ so $\mathbf{B}^T \mathbf{W} \mathbf{W}^T \mathbf{B} \preceq \mathbf{B}^T \mathbf{B}$. We thus have:

$$\sum_{i=1}^n l_i^\lambda(\mathbf{W}^T \mathbf{K} \mathbf{W}) = \sum_{i=1}^p \frac{\sigma_i(\mathbf{W}^T \mathbf{K} \mathbf{W})}{\sigma_i(\mathbf{W}^T \mathbf{K} \mathbf{W}) + \lambda} \leq \sum_{i=1}^n \frac{\sigma_i(\mathbf{K})}{\sigma_i(\mathbf{K}) + \lambda} = \sum_{i=1}^n l_i^\lambda(\mathbf{K}),$$

giving the lemma. □

D.2 Proof of Theorem 10: fixed sample size guarantees

We now prove Theorem 10, which gives the approximation and runtime guarantees for our fixed sample size algorithm, Algorithm 3. The theorem follows from the recursive invariant:

Theorem 20. *With probability $1 - 2\delta$, Algorithm 3 performs $O(ns)$ kernel evaluations, runs in $O(ns^2)$ time, and for any integer k with $s \geq 640k \log(2k/\delta)$ returns \mathbf{S} satisfying, for any \mathbf{B} with $\mathbf{B}\mathbf{B}^T = \mathbf{K}$:*

$$\frac{1}{2}(\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I}) \preceq (\mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda \mathbf{I}) \preceq \frac{3}{2}(\mathbf{B}^T \mathbf{B} + \lambda \mathbf{I}) \quad (40)$$

for $\lambda = \frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{K})$.

Proof. Assume by induction that after forming \mathbf{S}_0 via uniformly sampling, the recursive call to Algorithm 3 returns \mathbf{S}_1 satisfying:

$$\frac{1}{2}(\mathbf{B}^T \mathbf{S}_0 \mathbf{S}_0^T \mathbf{B} + \lambda' \mathbf{I}) \preceq (\mathbf{B}^T \mathbf{S}_1 \mathbf{S}_1^T \mathbf{B} + \lambda' \mathbf{I}) \preceq \frac{3}{2}(\mathbf{B}^T \mathbf{S}_0 \mathbf{S}_0^T \mathbf{B} + \lambda' \mathbf{I}). \quad (41)$$

where $\lambda' = \frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{S}_0^T \mathbf{K} \mathbf{S}_0)$. This implies that $\tilde{\lambda} = \frac{1}{k} \sum_{i=k+1}^n \sigma_i(\mathbf{S}_1^T \mathbf{K} \mathbf{S}_1)$ satisfies:

$$\begin{aligned} \frac{1}{2k} \left(\sum_{i=k+1}^n \sigma_i(\mathbf{S}_0^T \mathbf{K} \mathbf{S}_0) + k\lambda' \right) &\leq \tilde{\lambda} \leq \frac{3}{2k} \left(\sum_{i=k+1}^n \sigma_i(\mathbf{S}_0^T \mathbf{K} \mathbf{S}_0) + k\lambda' \right) \\ \lambda' &\leq \tilde{\lambda} \leq 3\lambda'. \end{aligned}$$

Combining with (41) we have:

$$\frac{1}{2}(\mathbf{B}^T \mathbf{S}_0 \mathbf{S}_0^T \mathbf{B} + \lambda' \mathbf{I}) \preceq (\mathbf{B}^T \mathbf{S}_1 \mathbf{S}_1^T \mathbf{B} + \tilde{\lambda} \mathbf{I}) \preceq \frac{9}{2}(\mathbf{B}^T \mathbf{S}_0 \mathbf{S}_0^T \mathbf{B} + \lambda' \mathbf{I}).$$

So, for all i , \tilde{l}_i^λ (which is computed using $(\mathbf{B}^T \mathbf{S}_1 \mathbf{S}_1^T \mathbf{B} + \tilde{\lambda} \mathbf{I})$ and oversampling factor 5 in Step 5 of Algorithm 3) is at least as large as approximate leverage score computed using \mathbf{S}_0 instead of \mathbf{S}_1 . If we sample by these scores, by Lemma 5 and Lemma 11:

$$\frac{1}{2}(\mathbf{B}^T \mathbf{B} + \lambda' \mathbf{I}) \preceq (\mathbf{B}^T \mathbf{S} \mathbf{S}^T \mathbf{B} + \lambda' \mathbf{I}) \preceq \frac{3}{2}(\mathbf{B}^T \mathbf{B} + \lambda' \mathbf{I})$$

which implies (40) since $\lambda' \leq \lambda$ since $\|\mathbf{S}_0\|_2 \leq 1$ so $\sigma_i(\mathbf{S}_0^T \mathbf{K} \mathbf{S}_0) \leq \sigma_i(\mathbf{K})$ for all i .

It just remains to show that we do not sample too many points. This can be shown using a similar reweighting argument to that used in the fixed λ case in Lemma 5. Full details appear in Lemma 13 of [CMM17]. When forming the reweighting matrix \mathbf{W} , decreasing $\mathbf{W}_{i,i}$ will decrease $\sum_{i=k+1}^n \sigma_i(\mathbf{W} \mathbf{K} \mathbf{W})$ and hence will decrease λ . However, it is not hard to show that the i^{th} ridge leverage score will still decrease. So we can find \mathbf{W} giving a uniform ridge leverage score upper bound of α . Let $\lambda' = \sum_{i=k+1}^n \sigma_i(\mathbf{W} \mathbf{K} \mathbf{W})$.

Using the same argument as Lemma 5, we can bound the sum of estimated sampling probabilities by $64 \log(\sum l_i^{\lambda'}(\mathbf{W} \mathbf{K} \mathbf{W})/\delta) \cdot \sum l_i^{\lambda'}(\mathbf{W} \mathbf{K} \mathbf{W}) \leq s/5$ by Fact 9. The runtime and failure probability analysis is identical to that of Algorithm 2 – the only extra step is computing $\tilde{\lambda}$ which can be done in $O(s^3)$ time via an SVD of $\mathbf{S}_1^T \mathbf{K} \mathbf{S}_1$. \square

Proof of Theorem 10. The theorem follows immediately since Theorem 20 guarantees that in the final level of recursion \mathbf{K} is sampled by overestimates of its λ -ridge leverage scores. The runtime bound follows from Theorem 20 and the fact that it is possible to compute $\mathbf{K} \mathbf{S}$ using $O(ns)$ kernel evaluations and $(\mathbf{S}^T \mathbf{K} \mathbf{S})^+$ using $O(ns^2 + s^3) = O(ns^2)$ additional time. \square